Exploring the complexity of risk interaction with success factors and success criteria in software projects

By

Fahad Harbi

A thesis submitted in partial fulfilment of the requirements of the degree of Doctor of Philosophy

School of Engineering

Faculty of Science and Engineering

University of Liverpool

April 2017

THIS THESIS IS DEDICATED TO MY PARENTS, WIFE, SON, BROTHERS, SISTERS AND ALL MY FAMILY AND FRIENDS

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Acknowledgements

First, I would like to thank Almighty God for his blessing in the completion of this research. Also, I would like to thank Dr Halim Boussabaine and Professor Huajiang Ouyang for their support, comments, guidance and constructive criticism during this research. Also, a special thanks to Mr Jack Carter-Hallam for all his help and support to overcome all my problems as PhD student. Also, I would like to thank my dear wife and son, who have been at my side from the first day: all the hard times that we have been through, may Allah bless you for all the support on this learning journey. I would also like to thank my family, my mother and brothers, Mohammad, Turki, Hamoud, Naif and Khalid, who have given me tremendous support to carry on with this research. I would also like to say thanks to my dearest friends who encouraged me to pursue my PhD study, especially Khalid Twlihan.

Finally I dedicate this research to my dearest uncles Jezza and Fahad, who died tragically this year – May Allah have mercy on their souls.

Abstract

In recent decades, information technology and software products have become an important part of all areas of life. Governments, businesses and individuals seek to set up software projects that, for example, provide support or provide a service or make our lives easier in other ways. This has resulted in the creation of a huge number of software projects. However, these projects often fail or are even cancelled before they start. Despite numerous studies to determine the reasons for this, software projects are still one of the riskiest types of project. This is because many of them contain a lot of overlapping parts that interact with each other, which increases their complexity. Therefore, the study of the complexity of interactions can lead to more understanding of such projects, which, in turn, even if it does not provide full understanding of a project's complexity, gives decision-makers the capacity to make the right decision for the success of a project.

This study explores the interactions that occur between the risk factors and success in software projects. An in-depth literature review was conducted to extract the most important risk and success factors of project software, and this research identified a list of 64 risk factors and 38 critical success factors taxonomised into three classes. Furthermore, four success criteria (cost, quality, time and scope) were identified for measuring success in software projects.

A mixture of methods from both qualitative and quantitative approaches has been used in this research. Also, the literature review has been used in order to review the relevant concepts related to the research topic. In addition, the literature review identified a gap in the research. Two surveys were used to obtain the necessary data in order to answer the research questions and objectives, and then the data were analysed using descriptive, statistical and network analyses.

The results of this research exploring the complexity of interaction have been produced in two parts. The first part is the constructs' correlation results; descriptive and statistical analysis of the data was generated by this method. In addition, this research explored the characteristics of the network that have been created by the interactions between risk factors, success factors and success criteria. Furthermore, it identified the most central and influential factors in those networks, and found that the development environment is the most important factor in a software project. In part two, a dependency matrix has been used to explore the interactions that occur within each success criterion as well as identifying the most controlling factors in each ego network. Effective project management has been identified as the most influential success factor in the cost, quality, time and scope ego networks; having an unrealistic budget was the most influential risk factor in the cost and scope ego networks, and resource insufficiency was the most influential factor in the quality and time ego networks. Furthermore, as well as the association between the factors, the contribution of each factor in predicting each success criterion has been analysed and discussed in order to give decision-makers as much information as possible to develop strategies and plans to raise the software project success rate.

Table of Contents

List of tables

List of Figures

Chapter 1: Introduction

Chapter 1: Introduction

1.1 Introduction

Software projects are considered one of the newest scientific fields that are being investigated. They face many problems. Thus, at the beginning of this chapter, the research context is explained. This is followed by an explanation of the research problem being investigated by the researcher, upon which the research questions are set. After that, the research aim and objectives are mentioned, and the chapter ends with a description of the research outline and the research structure.

1.2 Research context

Software development projects have seen a high level of investment during recent decades. This is because they have entered into all aspects of our lives. For example, governments are establishing software projects in order to assist them in the delivery of government services to their citizens, companies and institutions. Software products in the health sector have helped to increase the efficiency of healthcare. In the business sector, software products helped to organise businesses and increase profits. However, they have become the focus of a lot of managers, developers and researchers due to their high failure rates. According to Taherdoost and Keshavarzsaleh (2015b), the cost of software and IT project failure in the United States annually is about 150 billion dollars. Therefore, researchers are trying to understand the reasons for this failure. According to a report by GROUP (2013), just 39% of all software projects are delivered in accordance with the success criteria “quality, cost, scope and on time”. From this study, it is clear that nearly 61% of software projects partly or completely fail to achieve their set targets. These findings are supported by Jørgensen (2014), who noted that about 80% of all software projects are ether cancelled or fail. Most of the studies on software projects are centred around three main areas. Some investigate the risks facing software projects by identifying the risk factors or how they contribute to a project’s success (Colomo-Palacios et al., 2014, Vrhovec et al., 2015, Cerpa and Verner, 2009, Lu et al., 2010).

Other studies examine the success factors that help to increase a project’s chances of success (Ahimbisibwe et al., 2015, Stankovic et al., 2013, Nasir and Sahibuddin, 2011b). Most of these studies are based on the study of the impact of these factors on the key measures or criteria of success, “cost software quality, time and space”. Unfortunately, despite all the research and in spite of the amount of money that is being invested annually in risk management to improve the chances of success for software projects, project failure still occurs (Vrhovec et al., 2015). The interaction between these three aspects has not been investigated. As the concept that could be a major contributor to this failure is complexity, complexity has always been linked to projects in general. According Frame (2002), complexities cannot be separated from projects because projects are always complex and not fully understood, and complexity in software development projects is not an exception to that. This research believes that there is a need to uncover complexity between the latent variables that contribute to the success of software projects as the study of complexity leads to more understanding of a project. Even if it does not provide a full understanding of the project’s complexity, it does give decision-makers the capacity to make the right decision for the project’s success (Lu et al., 2015). Furthermore, according to Hill (1991), the ability to manage the complexity facing a project is the most important characteristic that project managers should have.

1.3 Research statement

Complexity is one of the most important topics to be studied in terms of projects in general and software projects in particular. This is due to its important and significant role in system development (Xia and Lee, 2005). Complexity might be considered one of the most important reasons for the failure of a huge number of software projects. Nguyen (2014), Calleam (2014) noted that nearly 35% of software projects fail because of the team's inability to understand the complexity of the project. The difficulty could be that, when the team realises that they have underestimated the project's complexity, it will often be too late to take steps to address this and ensure the project's success (Jørgensen, 2014). Another reason could be that it is difficult to study the complexity in all phases during a project (Nguyen, 2014). Furthermore, according to Petkova and Petkov (2015), "understanding software project complexity is an essential precondition for better IT project management". Complexity has been always closely associated with project success criteria. However, Lehtinen et al. (2014) stated that complexity arises because of the interaction between time, cost and quality. Furthermore, complexity occurs between risk and success factors associated with these criteria. According to Fitsilis (2009), researchers must carry out further study of the complexity of software projects in order to understand it, as current studies do not give enough focus to the phenomenon. He also added that, in order to understand the complexities, two important points need to be taken into consideration. The first is that in order to understand the complexities we must study the entire software project. The second is to understand the characteristics of the interactions between the system components or factors. This includes the importance of interaction and interdependency between the elements of the software project, as understanding those interactions could lead to building strategies and plans to predict and deal with these events, which in turn could help to increase the chances of software project success because the interaction between factors in a project has a huge impact on the project outcome (Nandhakumar, 1996). Based on that, this research believes that there is a lack of studies about complexity and no studies have uncovered the interaction between risk, success factors and success criteria in software development. Therefore, there is a need to uncover the complexity of interaction between these components, to give the decision-makers a better understanding about risk management of the software development project.

1.4 Research questions

Based on what has been previously reviewed concerning the problem and context of the research, this research attempts to answer the following questions:

- What are the most important risk factors and their classification in software development projects?
- What are the interdependencies (based on dependency matrix and constructs' correlation) between research constructs?
- What influence do risk and success factors have on success criteria?

Research value

This research believes that uncovering the interaction between risk and success factors and success criteria will help project decision-makers to develop appropriate strategies and the necessary steps to deal with events when they occur in a project. This is because risk management in projects has become one of the most important departments that contribute to the management of a project. Thus, uncovering the complexity of interactions will assist in the development of appropriate management plans to avoid these risks. In addition, uncovering the complexity of interaction helps to understand the impact of a factor on the other factors in a software project and the speed and strength of the impact. Furthermore, this research builds an interaction network between the software project components which shows the contribution of each factor. Furthermore the criteria ego networks revealed the interdependency between the software components. Accordingly, revealing the mechanism of the factors' interaction environment in a software development project helps to raise the project success rate taking into consideration the amount of knowledge it provided.

1.5 Aim and objectives

Aim:

This research aims to uncover the complexity of interaction between success criteria, risk factors and success factors in software development.

Objectives:

In order to reach that aim, this research translates it into the following objectives:

- To review, extract and classify risk and success factors in software development projects.
- To analyse the influence of risk on success factors and criteria using network metrics.
- To measure (capture) the interdependency between risk and success factors and criteria.
- To use ego topology to measure the influence of risk and success factors on success criteria.
- To isolate latent risks that influence success factors
- To isolate latent success factors and their associations with risk factors.

1.6 Outline of the research

Chapter one: this chapter provides an introduction to this research, including the context and the territory of the research. Also, the research problem has been mentioned; based on it, the research questions, aim and objectives have been developed. The chapter ends with the research outline.

Chapter two: this chapter highlights the well-known methodologies used in software development as well as providing a software definition and background knowledge about the software project and its characteristics.

Chapter three: this chapter focuses on the definition of risk as well as the relation between risk and software failure. It also provides background knowledge about risk management concepts. In addition, it explores the risk taxonomy used in related software development projects and identifies the risk factors.

Chapter four: this chapter focuses on the success concepts in a project via defining success and shows how success can be measured. . Furthermore, through the literature review this research reviews and identifies the success criteria as well as the success factors in software projects.

Chapter five: in this chapter, this research defines the main complexity concepts, system and characteristics. It also explores project complexity in relation to software projects. In addition, this chapter reviews the network definition, analysis, characteristics and centrality, and how networks can be likened to complexity.

Chapter six: explains the methodology of this research in detail. Furthermore, it outlines the data collection, analysis and discussion used in order to achieve the research aim.

Chapter seven: after data collection, this chapter performs descriptive statistical analysis using SPSS and Microsoft Excel. In addition, this chapter tests the perceptions of practitioners in detail.

Chapter eight: the risk, success and criteria characteristics are analysed and elaborated on in detail for each impact sub-network and the most central factors are identified.

Chapter nine: the interaction network is developed, analysed and elaborated on for all criteria. Statistical analysis is used to measure the centrality of the factors.

Chapter ten: the cost ego network topologies are applied and the data are analysed for each criterion, and the top centrality factors identified and elaborated on. In addition, the top central risk factor is isolated to study its influence on the success factors in the ego network. The same isolation methodology is applied to the success factors' influence on the risk factors.

Chapter eleven: the quality ego network topologies are applied and the data are analysed for each criterion, and the top centrality factors are identified and elaborated on. In addition, the top central risk factor is isolated to study its impact on the success factors in the ego network. The same isolation methodology is applied to the success factors' influence on the risk factors.

Chapter twelve: the time ego network topologies are applied and the data are analysed for each criterion, and the top centrality factors are identified and elaborated on. In addition, the top central risk factor is isolated to study its impact on the success factors in the ego network. The same isolation methodology is applied to the success factors' influence on the risk factors.

Chapter thirteen: the scope ego network topologies are applied and the data are analysed for each criterion, and the top centrality factors are identified and elaborated on. In addition, the top central risk factor is isolated to study its impact on the success factors in the ego network. The same isolation methodology is applied to the success factors' influence on the risk factors.

Chapter fourteen: this chapter discusses the main results and answers to the research questions.

Chapter fifteen: this chapter concludes the research; it explains the research contributions, provides recommendations for further research and highlights the research limitations.

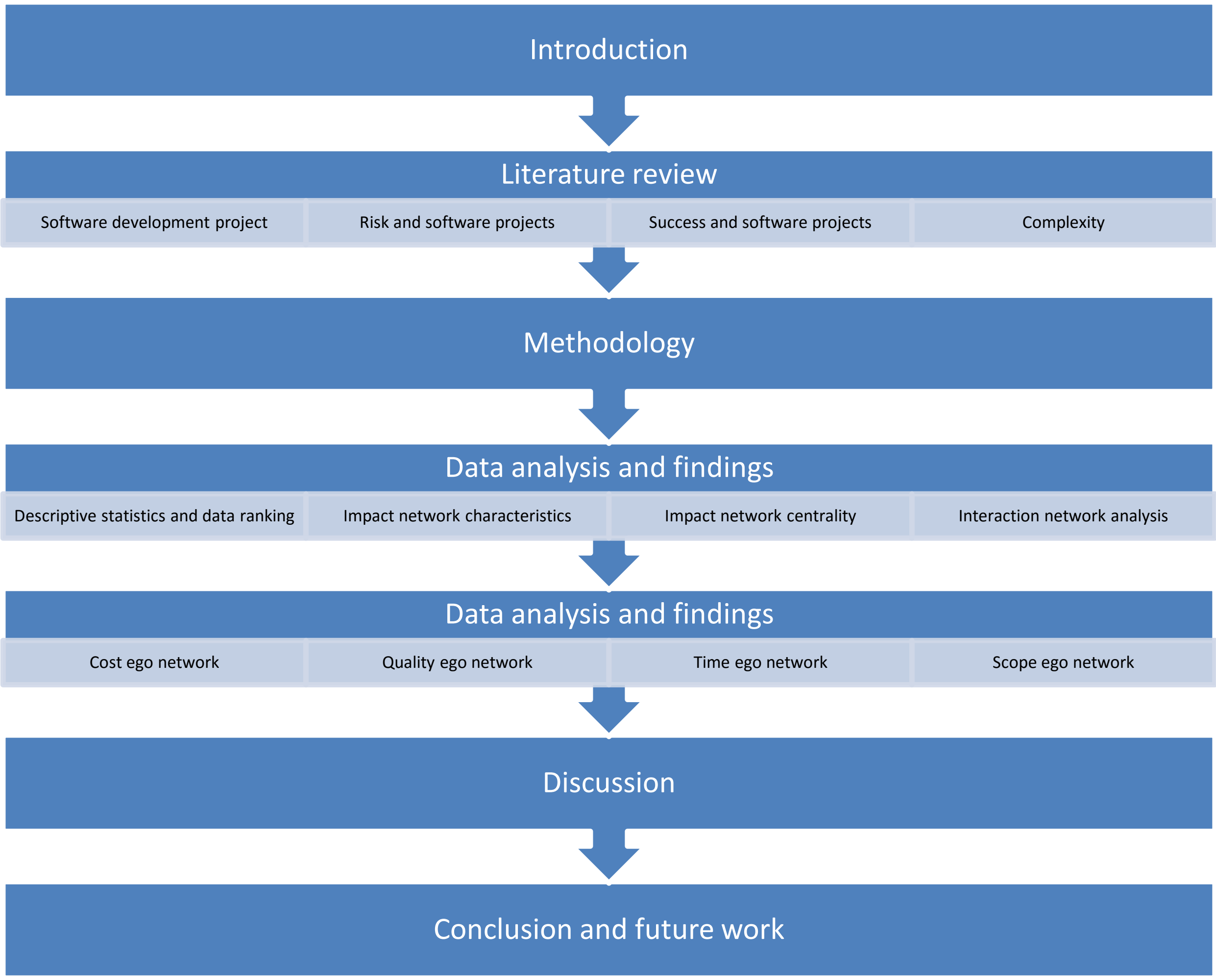


Figure 1-1 Outline of the research

1.7 Summary

There are many problems facing software projects. These problems often result in the failure of such projects, and complexity often plays an important role in these failures. This chapter has illustrated the research problem. In addition, this research aims to uncover the complexities facing software projects by investigating the interactions that occur between risk factors, success and success criteria. Each of these concepts will be reviewed in more detail in subsequent chapters.

Chapter 2: Software development projects

Chapter 2: Software development projects

2.1 Introduction

This chapter provides the literature review and background knowledge about the software project and its characteristics. It also reviews some of the well-known methodologies used in software development. The definitions of risk and software risk are also mentioned. Risk management and risk taxonomy are also discussed in this chapter.

2.2 Definition of software

According to the Cambridge Dictionary (2016), software is defined as “the instructions that control what a computer does; computer programs”. Also Oxford Dictionary (2014) defines it as "The programs and other operating information used by a computer”. From the two definitions, it is clear that the term software has always been related to controlling and solving problems or providing a service via a computer program. The computer program has been defined by the Oxford Dictionary (2016b) as “A series of coded software instructions to control the operation of a computer or other machine”. This shows that software development is considered to be a program or series of codes written using programing language in ordered to perform specific functions. Different approaches and methods can be used in software development to help to build a software project.

2.3 Software methodologies

Software development, like any development, has aspects that should be considered, such as the size of the project, the team, the complexity of the project, the nature of the institution, the product, the duration of the project, the techniques used and the requirements. This chapter will discuss some of the best-known software development methods and techniques.

2.3.1 *Capability Maturity Model integration (CMMI)*

Effectiveness and success are among the most important things that companies focus on in the establishment of a business. The success or otherwise of a project has become a big obsession for decision-makers in such organisations. For that reason, through the past decades several models and approaches have been developed to clarify and help to establish projects and increase their chances of success. However, these models and approaches often focus on a part of the project, or a certain type or size of organisation. No systematic approach has been developed to deal with projects that will be suitable for most organisations, regardless of the size of the project and the organisation. Several models have been created to deal with overall software development projects, such as the Software Engineering Institute's (SEI) Capability Maturity Model for Software (SW-CMM) (Constantinescu and Iacob, 2007).

With regard to CMMI, according to Shang and Shu-Fang (2009), “the government of the United States of America has funded with regard to software development a ‘set of best practices’ for Acquisition, Technology, and Logistics” to build on the “capability of maturity models to develop an integrated framework for supporting different capability maturity models and other related products”.

Integrated framework development in order to create continuous process improvement is one of the most important goals of CMMI. The determination of which parts of the software projects should be improved first, and which should be prioritised in an organisation, is also another benefit of using CMMI. Also, the organisation can use CMMI as a guide to determine the objectives and the procedures for the improvement of the software development process (Shang and Shu-Fang, 2009).

CMM integration was originally developed using three CMMs covering different aspects, such as product and service development and maintenance, by combining three models:

1. The Capability Maturity Model for Software (SW-CMM) v2.0

2. The Systems Engineering Capability Model (SECM)

3. The Integrated Product Development Capability Maturity Model (IPD-CMM)

CMMI also has a feature that enables the addition of any necessary models. Latterly, CMMI has consisted of a combination of four models:

1. Systems engineering
2. Software engineering
3. Integrated product and process development
4. Supplier sourcing.

   (Constantinescu and Iacob, 2007).

Capabilities at a lower stage provide foundations for a higher stage. Each development stage or maturity level identifies an organisation's software process capability (Paulk et al., 1993).

2.3.2 *Software development (SDLC)*

SDLC is the abbreviation for 'software development life cycle'. SDLC is used when talking about two important similar concepts. The first is about the software life cycle, while the second shows the structure and stages of the system used when developing software (Ruparelia, 2010). SDLC is a framework used to clarify the steps that are undergone in software development projects, beginning with, for example, the feasibility study, through requirements, analysis, design, development, test and then maintenance. There are a large number of models relating to the life cycle of the software development, which vary according to the size of the organisation, project type and duration of the project and so on. This research mentions some of the most common models used in the software development life cycle.

2.3.3 *Waterfall model*

The first software development model used sequel phases and was developed by Benington (1956). This model describes the software projects in nine phases (operational, analyses, operational specifications, design specifications, coding specifications, development, testing, deployment and evaluation) (Benington, 1956). Royce (1970) has simplified Benington's model into seven steps, where each step should be completed before the next step, although he also acknowledges that there will be some interaction between the steps, for example, during the testing phase, which will be looped through the other steps (Royce, 1970).

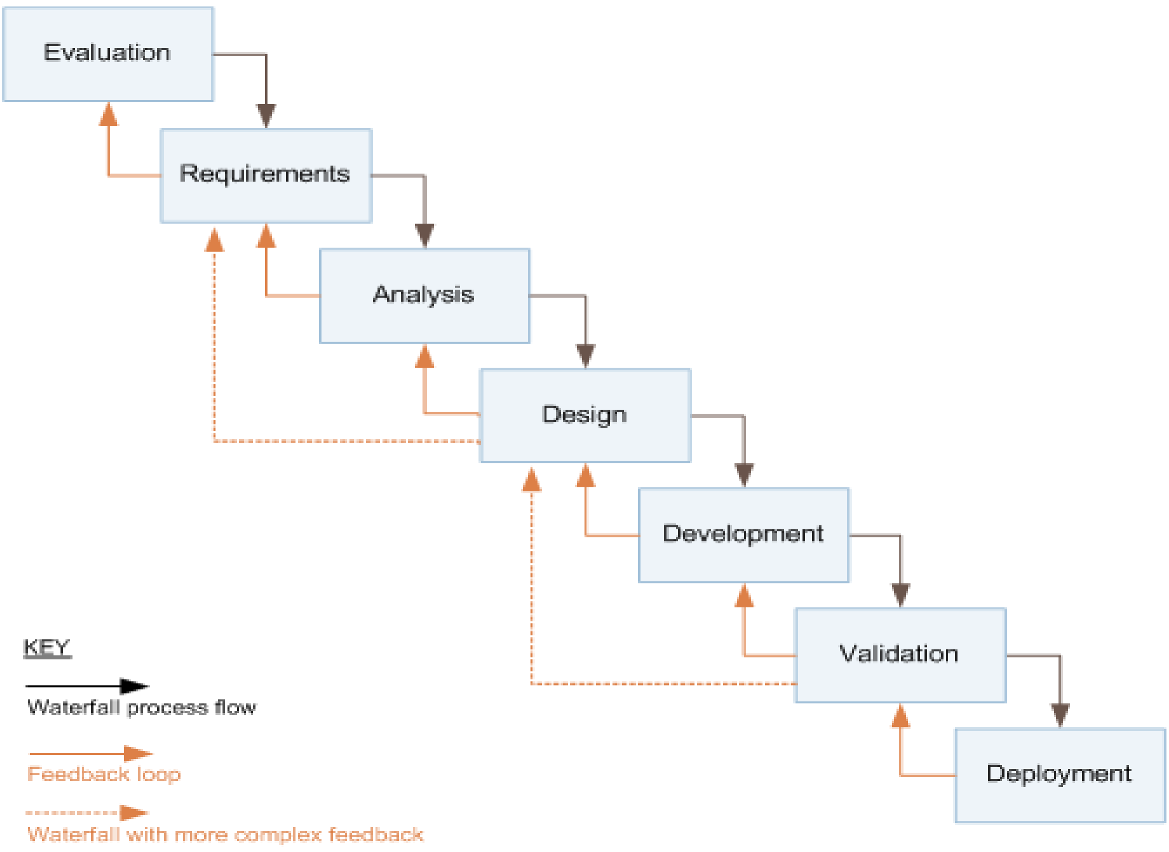


Figure 2-1 Waterfall model with Royce's iterative feedback Source: (Ruparelia, 2010)

The simplicity of the waterfall model is one of its major advantages because it can be easily applied by inexperienced teams. The objectives of maintainability as well as compliance can be achieved by taking into consideration the objective evidence, documentation and reviews. The waterfall model also increases the quality of the project significantly, because it provides strong support for the planning stage, as well as giving the management team greater ability to control the project (Doyle, 2010).

The inflexibility of the waterfall model is considered to be one of its major disadvantages. Because at the beginning of the project the team have to plan and take into consideration all the requirements needed for the software development, this makes it extremely difficult to respond to any change in the project. The system of working in sequential phases causes 'swelling' in the project. The lack of customer participation during the software project phases is considered a drawback. The timescale for projects using the waterfall model is longer compared to other projects using different models, which has resulted in such projects having a relatively high cost (CMS, 2005).

2.3.4 *Agile software development*

Agile, in its simplest description, can be described as an approach whereby the project is divided into mini projects, or a repeatable process of planning and developing. During the software development, teams like the business and project teams have to work together at the same time, which will noticeably shorten the duration of the project (Dearstyne, 2012).

Dearstyne (2012) has also noted that business users play an important role by interacting with the project during its development. By doing that, among other Agile principles, the software development becomes more fixable when there are any changes in the requirements.

2.3.4.1 *Agile recommendations*

Luz (2009) has made several recommendations so that Agile can be effective, as follows:

A. A small project and skilled programmers are considered important when applying Agile methods for software developments to be effective.

B. The presence of project team members in one place is a key reason for a project to be completed in the fastest time, as it provides a suitable environment in which to discuss the changes that occur during the project, and which could prevent it from achieving its goals and objectives.

C. The presence of members of the team in one place offers other benefits, such as being able to maintain the rhythm of work, enhance communication, and save time by not moving between work locations.

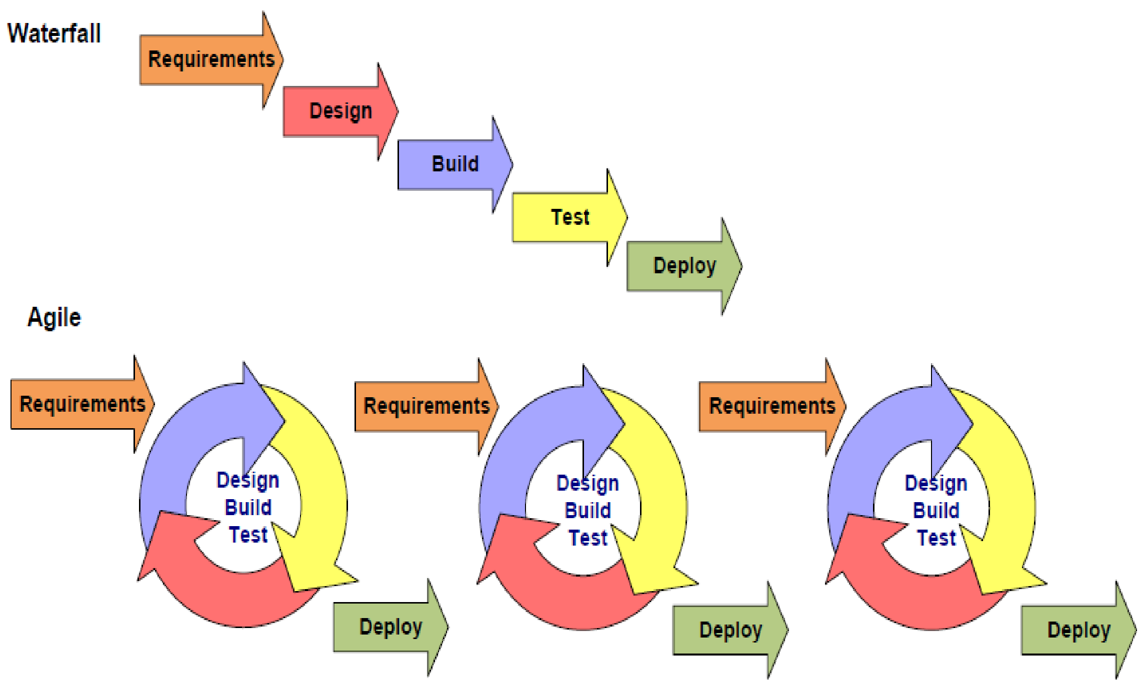


Figure 2-2 Waterfall vs. Agile Software Development Methods source:(Samra, 2012)

2.3.4.2 *Agile limitations*

The following table has been adapted from Misra (2007), except where indicated.

Table 2-1 When to use Agile and when NOT to use Agile source: (Wolfe, 2013)

When to use Agile	When NOT to use Agile
Rapid installation is a primary goal	Inexperienced project manager
Solid, experienced team	Newly-formed team
Strong project management	Necessity to conserve resources
Excellent project-related communications	Rigid team and organisation with tight control
Stable team composition	Rapid installation of the bulk of the system is a critical goal
Users are flexible and willing to work through many small implementations	Inexperienced system design personnel

Agile has many methods related to it, for example Extreme Programming (XP), Feature Driven Development (FDD), Dynamic Software Development Method (DSDM), Rapid Application Development (RAD), Adaptive Software Development (ASD), Crystal Clear, Rational Unify Process (RUP) and Scrum. These are the most popular Agile methods according to the Scrum Alliance (2011). Some of these methods will be mentioned below.

2.3.4.3 *Extreme Programming (XP)*

XP was developed in 1990 by a corporation called Chrysler. As an Agile method it focuses on continuous integration and developing; there are no separate analysis and design phases. Two programmers working together on one computer is a method emphasised in XP, as well as replacing the test stage with a stage called user acceptance (Anderson, 2004). In XP programming, project teams plan their time, cost and project scope based on three fundamental techniques: user stories, estimations based on past experience, and customer controls.

With XP, according to Munro (2003), the productivity of the programmers and experts has increased and the communication between programs has improved efficiently. Cusumano (2007) points out that user stories, estimations based on past experience, and customer controls are essential when planning cost, time and scope.

Repenning et al. (2001) noted that there are two limitations linked to software development in XP. The first limitation is related to the merging of programmers and domain experts, wherein it is difficult to provide this in a work environment that relies on distribution tasks. The second limitation is that the parallelism technique used in XP has not given an obvious or guideline explanation with regard to more than one team working on more than one task or component.

2.3.4.4 ***Scrum***

The name Scrum as used in this field first came from a rugby game in Japan in the 1980s (Koch, 2010). Scrum is an Agile methodology because the Agile manifesto has provided the principles and rules used in it (Luz et al., 2009). These rules clarify that stakeholders should be involved in the process of software development with the development team. The Scrum model emphasises that software development should adapt to change during the life cycle. Transparency between the development team and the business stakeholders is an important principle in Scrum. As an Agile method, the team being in one physical place gives the project the advantage of good communication and quick adaptation to any strategic change (Luz et al., 2009; Wolfe, 2013).

Scrum organises software projects into three levels: sprints, releases and products. It explains that releases contain collections of the other two levels: sprints and products (Anderson, 2003). The development team in Scrum develops software in a fixed and agreed time, usually no more than four weeks, by dividing the work into a sequence of sprints (Ionel, 2008). According to Ruparelia (2010), there are two aspects that should be taken into consideration when using Scrum methods. First, because of the need for the team to be in one place, Scrum is more suitable for small projects. Second, small or medium teams are recommended when using Scrum as it is challenging to applying the central command structure amongst a big team (Ruparelia, 2010).

Table 2-2 Summary of XP and SCRUM principles source: (Wolfe, 2013)

XP	SCRUM
Planning Around User Stories Plan the project work based on three techniques: Create stories that capture the functionality or features the customer wants, generate estimates from the developers of how long it will take to create each of these features, and then ask the customer to prioritise which features they want in which order.	**The Sprint** A time-boxed period, which generally varies from 2 to 4 weeks, where the team must build and deliver some amount of working software.
Small Releases of Functionality Build the product around user stories in small increments of functionality and then show the product to the customer or representative users as often as possible.	**Scrum Master** The Scrum master is responsible for the Scrum process, for implementing it in the project and for ensuring that everyone in the project follows and respects the set of practices and rules.
System or Project Metaphor Assign an organising metaphor to each project in order to provide common focus and to help the team and the customer understand what the project is about and what is most important.	**Team Member** Team members are responsible for developing the project. They have to work collectively, managing and organising themselves.
Simple Design Team members focus on building only what user stories suggest are the customer's immediate needs in the simplest way. There is an assumption that future requirements will change unpredictably, therefore eliminating the need for future detailed requirements.	**The Product Owner** The product owner is responsible for managing the project budget, release plans and list of project requirements, as well as providing the team with all necessary information related to his/her expectations of the software.
Refactoring Product designs and other subsystems or systems will have continuous revisions in order to accommodate new functionality as well as to eliminate redundant code.	**Product Backlog** 5 – 15 high-level features the product owner hopes to have in this release.

Pair Programming Team members to write code in pairs with two people sitting at one computer. This way one person is always reviewing what the other person is writing to catch potential errors, as well as to spread knowledge of how the product works at a detailed level.	**Burndown Chart** A burndown chart shows the progression of a team through a set of work. The remaining work appears to 'burn down' as the team completes more and more over the set of time viewed. The goal of the team is to complete all planned work by the end of the period so that the final snapshot shows nothing left to do.
Collective Product (Code) Ownership A team member can change any part of the system in any module. Team members take individual responsibility for tasks and delivery dates, but they share ownership of the actual code.	**Sprint Backlog** The team plans the strategy to achieve the goal of the sprint, discussing and defining the necessary tasks to be accomplished during the iteration.
Continuous Integration XP team members synchronise their changes and merge or 'integrate' them into the evolving code base at least daily, as well as run their tests before and after the integration exercise. If any problems occur, they must fix the code immediately. In this way, the evolving product is always supposed to work.	**Spring Planning Meeting** The product owner presents the highest priority items and the team questions him/her about how the item should work, and then the team chooses the functionalities that will be developed in the next interaction according to their time estimation.
Minimal Overtime Strongly discourage programmers from working overtime.	**Daily Scrum** A 15-minute standup meeting where each team member answers three questions: "What have you done since the last Daily Scrum?", "What will you do before the next Daily Scrum?" and "What impediments are in your way?"
On-site Customer Representative To avoid mistakes and wasted time, teams work alongside the customer for a custom project or with representatives of the customer, such as product managers, for more packaged or standardised products.	**Sprint Review** In the first half of a Sprint Review, the team members present the work done to the product owner and other stakeholders. The second half of the Sprint Review is where the team discusses the process and techniques used in the past Sprint and how they can improve their process.

2.4 Software development projects

Software development started as part of the computer-based industry, where it was focused on solving problems relating to service delivery and data analysis. But over the years the number of software projects increased rapidly, and the use of software methodology to achieve a specific goal became known as a software project. The main rule of these methodologies is to increase a software project's probability of success (Špundak, 2014). Moreover, each methodology has been used to mitigate specific risk factors. For example, the waterfall model avoids the risk of ambiguity in user requirements as the requirement and analyses stages have to be done before the coding and design development (Hijazi et al., 2012). On the other hand, this could be a risk factor if there are continuous requirement changes, as in the waterfall model the development stage requirements cannot be changed as they have been specified in a previous stage. In comparison, the agile methodology addresses this risk factor due to its flexibility in allowing the requirements to be changed, as has been mentioned earlier in this chapter. Furthermore, many researchers have studied software projects from different perspectives without focusing on specific methodology. For example, researchers have studied the software project from a risk and risk management viewpoint (Nakatsu and Iacovou, 2009, Abdelrafe and Burairah, 2013, Lu et al., 2013, Hu et al., 2013), which has resulted in software projects becoming an industry in themselves. This type of project needs to be managed as the humans responsible for specifying requirements, analysing, designing, developing, implementing, testing and delivering the product to the client need to be managed. Project management can be defined as the "application of knowledge, skills, tools and techniques to project activities to meet the project requirements" (Too and Weaver, 2014). Software project management shares many management concepts and techniques with other types of projects like constructional projects. But one of the main issues that face a software project's management team is that it is hard to show the customer the final product before it is completed.

Contrast this with the situation in construction management, where blueprints of the building, for example, will give a good idea of the final product's look help to manage the client's expectations (Cadle and Yeates, 2004). Complexity, multidependent and ineffective application of management methods, technics and practices have been identified as being among the main causes of software project failure over the years (Trendowicz and Jeffery, 2014a).

2.4.1 *Software project characteristics*

Software projects are not entirely like other projects; they have several characteristics that make them unique. Technological advances are rapidly changing, which in turn is affecting the software industry and posing a threat to the software business – these are some of the characterises that make software projects different to other projects (Stankovic et al., 2013). Furthermore, some of the noticeable characteristics of software development projects are complexity, flexibility, no constraints and invisibility, as there are many components working and affecting each other in a software project, which makes it difficult to understand or predict the outcome. Also, in most software projects, there is no one true answer to a problem, as it could be solved in many ways, which makes software projects flexible. Similarly, due to their flexibility, software projects are one of the least constrained types of project. However, on the other hand, software invisibility is one of the most challenging characteristics of software projects, as the outcome of the project is hard to visualise and present to the client (Niazi and Babar, 2009, Altahtooh and Emsley, 2014a, Butler et al., 2004). Ruhe and Wohlin (2014b) found that there are several characteristics that make software projects different to and more difficult to manage than other projects, as can be seen in the table below.

Table 2-3 Software project characteristics source: (Ruhe and Wohlin 2014)

• Software is an intangible product. • Software is a cognitive and human-based development process that requires sharing of documents. • There is a higher degree of uncertainty in the project and product scope. • Communication and coordination within software teams and with project stakeholders often lacks clarity. • The intellectual capital of software personnel is the primary asset of software projects and organisations. • There is a degree of change of requirements in the course of the software project. • The creation of software requires innovative problem solving to create unique solutions. • Initial planning and estimation of software projects is challenging because these activities depend on requirements that are often imprecise or based on information that is not available. • The development and evolution of software-intensive systems is challenging because of the high complexity of software based on the enormous number of logical paths in program modules and all the combinations of interface details. • Exhaustive testing of software is impractical because of the time and related complexity constraints. • Software development often involves interactions of different vendor products and interfaces with other software. • Software security is a large and growing challenge. • Objective measurement and quantification of software quality is difficult. • Learning and knowledge creation in software development is more difficult because processes, methods and tools are constantly evolving. • The execution of software is platform-dependent and is often an element of a system consisting of diverse hardware, other software and manual procedures.

2.5 Summary

This chapter has defined software as well as illustrating the definition, style and characteristics of a number of methodologies used in software development projects. It also sought to show what a software project is and what software project characteristics are.

Chapter 3: Risk and software projects

Chapter 3: Risk and software projects

3.1 Introduction

During project life cycle there are factors that, if they occur , may risk the success of the project and result in its failure. This chapter defines risk and investigates the situation relating to risk and software project failure. In addition, this research will also discuss a number of risk management concepts. The chapter will conclude with a review of the classifications of the risks identified as well as the most important risks that face the software project.

3.2 Definition of risk

According to the Oxford Dictionary (2016b), risk can be defined as “The possibility that something unpleasant or unwelcome will happen”. Fairley defines risk as “The probability of incurring a loss or enduring a negative impact”. Similarly, the ISO defines it as the “effect of uncertainty on objectives” (ISO, 2009). Also, Kontio (1997) notes that “The term risk in its general meaning is defined as a possibility of loss, the loss itself, or any characteristic, object or action that is associated with that possibility”.

In accordance with Fischhoff et al. (1984), risk is the central theme in the supervision of many technologies and activities. An open and accepted definition of the word ‘risk’ is necessary in order to make its management successful. According to Oliver (2012), a common definition of risk implies that through preventative action risk may be avoided, and it is the possibility or threat of liability, loss, damage, injury or any other harmful occurrence that is caused by external or internal vulnerabilities. To conclude, although there are many risk definitions and approaches, their similarity relies on the fact that risk is an event that might or might not occur during a project, and, if it did happen, it would be likely to have an impact on the project.

3.3 Software risks and failures

Software development projects are considered one of the most risky types of project nowadays. There are a considerably high number of software project failure (Bannerman, 2008, Altuwaijri and Khorsheed, 2012). Risk may vary depending on the type of project. Each project type has its own methods and characteristics that distinguish it from other types of project. Many software projects are not delivered on time, whilst others exceed their initial planned cost or face problems in quality and scope. According to Islam (2009), risk in software can be defined as the probability of an event happening during the project life cycle, such as budget or schedule over-runs, customer dissatisfaction, poor quality, and passive customer involvement due to an undesirable event and its consequences. Ye Tao (2009) notes that termination, delays in schedule and over-run of resources are typical undesirable and prohibited events which can affect software development projects. Using untested technologies, system requirements, system architecture, performance of systems, non-core activities and aspects of the organisation are also well-known aspects that pose risks in software projects. According to Charette (2005), software projects are considered high-risk activities. About half of all software projects fail for a variety of reasons, or are not delivered on time. KPMG (2005) note that this failure is not exclusive to a particular country or continent, but is considered at the global level in both the public and private sectors (Bannerman, 2008). Some reports have estimated this failure to be between 65% and 70% in ICT projects (Self and Aquilina, 2013). A report by one study group has estimated that only 39% of software projects were delivered on time, within the cost, and thus can be considered as successful projects (GROUP, 2013, Raith et al., 2013). This shows that more that 60% are considered as failures. About one-fifth of software projects failed completely or in extreme cases were cancelled'. There are many example of software project failure. One of the early software project failures was TAURUS: it was designed to transfer the London Stock Exchange to an automated system for settling stock transactions, but was cancelled in 1993 due to management issues and an unrealistic budget, which caused the project to overrun the budget (Charette, 2005b). Another example is the Sydney Water Board software project, which failed as a result of unrealistic cost, unrealistic schedule, inadequate requirements and change management (Verner et al., 2008a). Another big example is the U.S. Healthcare.gov website, as the project failed to meet its original budget, according to Anthopoulos et al. (2016), some of the risk factors that caused the software and the website to fail are: “Unrealistic requirements, No particular budget and cost planning and Complex architecture design”. Managing risks in software projects is slightly different to other projects because there are risk factors that are related to software development only, like designing software with many coding styles (Tomer, 2014). In order to increases the software project success rate, these risks and how they affect the project should be studied. Many studies have studied risk in software projects from many aspects but most of them mention the risk factors as a list and the main focus is on how they affect the project (Colomo-Palacios et al., 2014, Vrhovec et al., 2015, Cerpa and Verner, 2009, Lu et al., 2010). Thus, important aspects of the software project have not been widely understood. Furthermore, the interaction and relationship between the risk factors and the factors that help to minimise those risks impact on the software project have not been studied. It is important to understand how the risk and success in software development projects can be improved and how the risk can be approached (Lehtinen et al., 2014).

3.4 Risk management and software projects

3.4.1 *Risk management*

Risk management has two main concepts attached to it. The first is analysis, as the first concept in risk management is the analysis of risk in a project: to identify it, understand how it works and work out the dimensions relating to exposure to this risk in the project. The second is how to control this risk in terms finding ways, approaches, models and tools to reduce the impact of this risk or take precautions when it happens (Charette, 1989, Boehm, 1991). Also, Powell and Klein (1996) noted that one of the most important benefits of risk management is to maximise the potential advantages of the situations that occur during a project by identifying and preparing for actions to reduce these risks (Powell and Klein, 1996). Another view of risk is that in order to manage a risk it is important to understand that two of its most important characteristics are uncertainty and loss (Iranmanesh et al., 2009). Uncertainty, or what also could be known as complexity in risk management terms, refers to a situation where an event or events could occur but the outcomes of those events are difficult to predict. Furthermore, those events in some cases are not totally understandable from many aspects, like how they impact on the project, interact with other elements of the project, and what overall effect they will have on the project, which will be discussed in more detail in the next chapter.

Loss can be defined as the outcome of a risk event that occurs before, during or after the project has been completed, where the outcome of this event is unwanted and unwelcome. Also, the Software Engineering Institute (SEI) defines loss as " the impact to the project which could be in the form of diminished quality of the end product, increased costs, delayed completion, loss of market share, or failure" (Rane et al., 2011). Based on that, another view of risk management is to develop a plan to manage risk if it occurs during the project life cycle, where the risk can either be avoided, controlled or accepted.

3.4.2 *Risk management paradigm*

There are many phases and processes related to risk management. According to Higuera and Haimes (1996), risk management is a continuous process. In this process of managing risk in software projects, they have identified six stages, as presented in the following table:

Table 3-1 Risk management paradigm

Identify	The first step in managing risk is to identify those risks, as any ambiguity in identifying the risks could have an impact on the project cost, time, quality and scope (Wang et al., 2010). Furthermore, the identification could be through a list of potential factors and their areas (Bruckner et al., 2001).
Analyse	Project managers and decision-makers use risk analysis to give them an understanding and information about certain situations that they could face in their projects, as this information can help them to avoid project failure (Wallace et al., 2004, De Bakker et al., 2010).
Plan	In order to take action after identifying and analysing risk in the project, a pan should be available to decision-makers. This plan could contain certain precautions or actions or accept the outcome of the risk factors that occur during the project.
Track	Tracing risks is all about monitoring the risk related to the project. Decision-makers should always be aware of the risk statues, what the action plans to manage this risk are and how they are working. It is important to understand that risks happen during all stages of a project.
Control	Risk control is the main tool in risk management action, as an action plan for the most important risk factors should be implemented (Kontio et al., 1998). Risk control should be a part of the project management process. This control should use the models and approaches identified in the planning stage.
Communicate	Higuera and Haimes (1996) noted that the importance of risk communication is it "Lies at the centre of the model to emphasise both its pervasiveness and its criticality. Without effective communication, no risk management approach can be viable. While communication facilitates interaction among the elements of the model, there are higher-level communications to consider as well. In order to be analysed and managed correctly, risks must be communicated to and between the appropriate organisational levels".

3.5 Risk taxonomy

The taxonomy of risk is considered one of the most important pillars in dealing with risk. It is important that, after risks are identified, they are distributed to certain groups who can share links or have common characteristics. This taxonomy could provide a greater opportunity to deal with and to find solutions or procedures to deal with those risks. Furthermore, it can help the researcher with data analysis. It also has the ability to set repeatable and applicable risk frameworks. Over the years, there have been many attempts to classify risk in software projects and IT projects from different aspects for example, Jiangping et al. (2013) in their research about “Risk Factors of ICT” classified risks into three risk categories, technology management, project management and dynamic capability, followed by nine subcategories. Other researchers have classified risk in simpler ways, such as Anand and Chopra (2012), who identified 15 factors in their Decision Support System for Software Risk Analysis, classifying risks into two main classes, input factors and output factors. Similarly, with regard to software supply chain risks, Du et al. (2013) mentioned two main risk classifications, external risk and internal risks, and added eight subcategories under these classifications. Clarke and O’Connor (2012) classified risks into wider classifications in their research about the software development process, using eight classes: personnel, organisation, requirement, operation, application, management, technology and business. Lu and Yu (2012) categorised risks into five main categories related to software development projects. In software process management risk factors, Li et al. (2012) found that requirement, user, developer, project management, technical risk and organisation & environment risk were the most suitable classifications for risk. Likewise, in their research about quality and risk in software development, Sarigiannidis and Chatzoglou (2013) classified risks into six dimensions: user, requirements, project complexity, planning & control, team and organisational environment.

As can be seen from this short narrative, the classification changes based on the objective and area of the research investigation. As the main objective of this research is to uncover the complexity of risk and success factors interaction in software development projects, it is believed that the taxonomy from the Software Engineer Institution (SEI) is the most suitable to help the researcher to achieve this aim. More details and reasons behind choosing this taxonomy will be discussed in the next section.

The following table summarises the classifications that have reviewed by this research. Each classification has been mentioned as well their sub-classes.

Table 3-2 Risk classifications in IT and software developments reviewed in the research

<table>
<tr><th>Reference</th><th>Classification</th></tr>
<tr><td>Jiangping et al. (2013)</td><td>
<table>
<tr><th>1. Technology management</th><th>2. Project management</th><th>3. Dynamic capability</th></tr>
<tr><td>1.1 Technology selection</td><td>2.1 Strategic positioning</td><td>3.1 Organisational learning</td></tr>
<tr><td>1.2 Technology utilisation</td><td>2.2 HR management</td><td>3.2 Organisational agility</td></tr>
<tr><td>1.3 Technology protection</td><td>2.3 Marketing</td><td>3.3 Resources allocation and integration</td></tr>
</table>
</td></tr>
<tr><td>Du et al. (2013)</td><td>
<table>
<tr><th>1. External risk</th><th>2. Internal risk</th></tr>
<tr><td>1.1 Natural disaster</td><td>2.1 Participants</td></tr>
<tr><td>1.2 Political factor</td><td>2.2 Software components</td></tr>
<tr><td>1.3 Economic factor</td><td>2.3 Operation and maintenance</td></tr>
<tr><td>1.4 Social factor</td><td>2.4 Supply chain logistics</td></tr>
</table>
</td></tr>
<tr><td>(Verner and Abdullah, 2012)</td><td>
<table>
<tr><td>1. Complexity</td><td>2. Organisational environment</td></tr>
<tr><td>3. Contract</td><td>4. Planning and control</td></tr>
<tr><td>5. Execution</td><td>6. Scope and requirements</td></tr>
<tr><td>7. Financial</td><td>8. Team</td></tr>
<tr><td>9. Legal</td><td>10. User</td></tr>
</table>
</td></tr>
</table>

(Clarke and O'Connor, 2012)	1. Personnel	2. Organisation
	3. Requirement	4. Operation
	5. Application	6. Management
	7. Technology	8. Business
(Anand and Chopra, 2012)	1. Input Factors	
	2. Output Factors	
(Khatavakhotan and Ow, 2012)	1. User side	2. Process maturity
	3. Technology	4. Maintainable
	5. Environment	6. Subcontract
	7. Project nature	8. Security – confidential
	9. Project plan	10. Personal and staffing
Lu & Yu (2012)	1. Organisation function risk	
	2. Developing technology risk	
	3. personnel system risk	
	4. Resources integration risk	
	5. System requirement risk	
Sarigiannidis and Chatzoglou (2013)	1. User	
	2. Requirements	
	3. Project complexity	
	4. Planning and control	
	5. Team	
	6. Organisational environment	
Li et al. (2012)	1. Requirement risk	
	2. User risk	
	3. Developer risk	
	4. Project management risk	
	5. Technical risk	
	6. Organisation & environment risk	
(Verner et al., 2014)	1. Outsourcing rationale	
	2. Software development	
	3. Human resources	
	4. Project management	
(Samantra et al., 2014)	1. Strategic risk	2. Environmental risk
	3. Business risk	4. Information risk
	5. Technical risk	6. Managerial risk
	7. Financial risk	8. Relationship risk
	9. Legal risk	10. Time management risk
	11. Operational risk	
(Hoodat and Rashidi, 2009)	1. Software requirement risks	
	2. Software cost risks	
	3. Software scheduling risks	
	4. Software quality risks	
	5. Software business risks	
Keil et al. (2008) and Schmidt et al. (2001a)	1. Corporate environment	2. Sponsorship/ ownership
	3. Sponsorship/ ownership	4. Relationship management
	5. Relationship management	6. Project management
	7. Project management	8. Scope

	9. Scope	10. Requirement
	11. Requirement	12. Funding
	13. Funding	14. Scheduling
	15. Personal	16. External dependencies
	17. Staffing	18. Planning
	19. Technology	20. Others
Han and Huang (2007)	1. Team risk	
	2. Organisational environment risk	
	3. Requirements risk	
	4. Planning and control risk	
	5. User risk	
	6. Complexity risk	
Williams et al. (1999)	1. Requirement	
	2. Design	
	3. Code and Unit Test	
	4. Integration	
	5. and Test	
	6. Engineering Specialties	
	7. Development Process	
	8. Development System	
	9. Management Process	
	10. Management Methods	
	11. Work Environment	
	12. Resources	
	13. Contract	
	14. Program Interfaces	

3.6 Software risk classification in this research using the SEI's risk taxonomy

The primary objective of this research is to find the complexity of risk in software development, and one of the most important classifications used in determining risk is the SEI's Risk Taxonomy. Therefore, this research uses this classification when searching for risk for several reasons.

First, this method has been designed by a well-known organisation specialising in software development, which is called the Software Engineer Institution (SEI). The method was developed in 1993 (Carr et al., 1993), and has been determined as much as possible without neglecting the risk of any side and through the use of several methods to identify risks, such as ‘checklist’ or expanded ‘radar screen’ (Gallagher et al., 2005). The taxonomy was first published in Taxonomy-Based Risk Identification (Carr et al., 1993). It has subsequently been republished in a different format, but with no change in content, in the Continuous Risk Management Guidebook (Dorofee et al., 1996) and in SRE Method Description (Version 2.0) & SRE Team Member’s Notebook (Version 2.0) (Williams et al., 1999, Williams et al., 2004). Also, according to Na et al. (2007), this taxonomy is still widely used by many organisations.

Second, one of the most important things that prompted this research to adopt this method is that it has been tested across several sectors (Carr et al., 1993). This classification has proved to be according Dougherty and Papadopoulos (2005)“useful, usable, and efficient” as it has been tested in identifying risks in software development projects in both the government and private sectors.

Third, since its publication, this method has been used by many researchers (e.g. Na et al., 2007; Higuera & Haimes, 1996; Gallagher et al., 2005; Kendall et al., 2007; Kontio, 1997; Maniasi, 2006; Freimut, 2001; Choetkiertikul & Sunetnanta, 2012; Menezes et al., 2013; Sonchan & Ramingwong, 2014). Also, it has been used by many organisations. The MBA and Arias (2011) have noted that the SEI method has been used by the Xerox Corporation, the State of Pennsylvania, Computer Sciences Corporation, the US Army, NASA and the US Air Force.

Fourth, one of the most important features of the taxonomy is that it takes into account two important concepts: the theoretical part (literature) and the testing part: “40 field tests were conducted with a broad range of software developers before coming up with a good interviewing technique for risk identification the Taxonomy-Based Risk Identification method” (Williams, 2008).

Fifth, according to Shahzad and Al-Mudimigh (2010), the taxonomy emphasises the need to use questionnaires in the identification of risk factors, which in turn meets one of the methodologies used in the data collection for this research.

Sixth, according to Higuera and Haimes (1996), the SEI's taxonomy covers the three aspects related to software project development: product engineering, development environment and program constraints.. Even this comment by Higuera and Haimes is supported by the fact that this taxonomy has not been changed and it is still used in many organisations, even approximately a decade after it was released.

3.6.1 *SEI's taxonomy*

In the SEI's taxonomy, risks are classified into classes, elements and attributes, as shown in Figure 3-1. According to Higuera and Haimes (1996) A project is divided into three main classes, as follows:

"1. Product engineering: the technical aspects of the work to be accomplished.

2. Development environment: the methods, procedures and tools used to produce the product.

3. Program constraints: the contractual, organisational and operational factors within which the software is developed, but which are generally outside of the direct control of the local management."

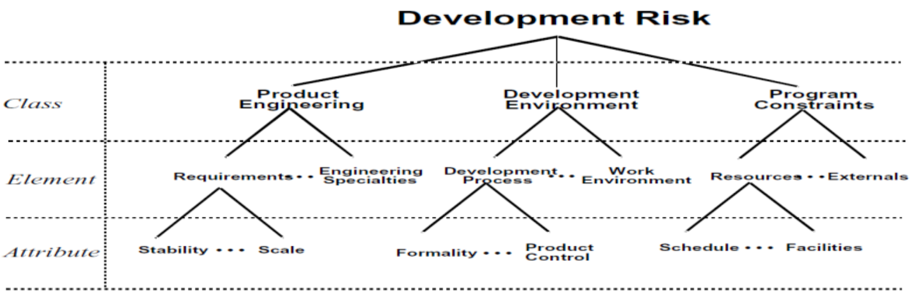


Figure 3-1 SEI Taxonomy Structure source: (Higuera and Haimes, 1996)

The fallowing has been taken from the SEI taxonomy, except where indicated:

3.6.1.1 *Product engineering*

As mentioned above, product engineering is more focused on the technical aspect of the project, where attention is paid to analysis of the software and hardware and documentation of the final product. This process contains five elements: requirements, design, code and unit test, integration and test, and engineering specialties. Product engineering refers to the system engineering and software engineering activities involved in creating a system that satisfies specified requirements and customer expectations. These activities include system and software requirements' analysis and specification, software design and implementation, integration of hardware and software components, and software and system tests. The elements of this class cover traditional software engineering activities. They comprise those technical factors associated with the deliverable product itself, independent of the processes or tools used to produce it or the constraints imposed by finite resources or external factors beyond program control. Product engineering risks generally result from requirements that are technically difficult or impossible to implement, often in combination with inability to negotiate relaxed requirements or revised budgets and schedules; from inadequate analysis of requirements or design specification; or from poor quality design or coding specifications. The list of important risk factors in the product engineering class can be seen in the following table:

Table 3-3 Product Engineering risk factors

Code	Product engineering risk factors	References
R1ENG	Unclear customer requirements	Sonchan and Ramingwong (2014) Asif et al. (2014) Wallace et al. (2004) Sonchan and Ramingwong (2015) Hu et al. (2013) Sarigiannidis and Chatzoglou (2014) Sipayung and
R2ENG	Unable to meet user requirements	
R3ENG	Lack of technical skills	
R4ENG	Technical complexity	

R5ENG	Low software performance	Sembiring (2015) Reifer (2002) Dey et al. (2007) Fraser and Arcuri (2014) McManus (2012) Hijazi et al. (2014b) Shahzad and Al-Mudimigh (2010) Bazaz et al. (2012) Moorthy et al. (2013) Koopman (2011) Kirk and MacDonell (2009) Nakatsu and Iacovou (2009) Addison and Vallabh (2002) Moorthy (2014)
R6ENG	Requirement creep (Constant changes in the requirements)	
R20ENG	Understanding problems of customers	
R21ENG	Understanding problems of developers	
R28ENG	Inappropriate design	
R29ENG	Inappropriate technology	
R31ENG	Improper marketing techniques	
R33ENG	Size of the project	
R34ENG	Incompatible development environment	
R36ENG	Problems in testing tools	
R44ENG	Less reusability	
R45ENG	Excessive error detection	
R46ENG	Architecture complexity	
R48ENG	Inconsistent coding style	
R49ENG	Lack of adequate security technologies (e.g., firewalls, encryption, etc.)	
R52ENG	Insufficient consideration of reliability/availability	
R53ENG	Insufficient consideration of system reset approach	
R57ENG	Use of cheap tools (software components, etc.) instead of good ones	
R58ENG	Insufficient consideration of security and safety	
R61ENG	Lack of mechanism for validation and verification	
R62ENG	Unclear or misunderstood scope/objectives	

3.6.1.2 *Development Environment class*

This class is more concerned about the project environment in which the software product is being developed. This class contains five elements: development process, development system, management process, management methods and work environment. The development environment class addresses the project environment and the processes used to engineer a software product. According to Sonchan and Ramingwong (2014), the majority of risk factors in software projects are classified under the development environment, and this research has made the same finding, as can be seen in :

Table 3-4 Development Environment risk factors

Code	Development Environment risk factors	References
R7DEV	Inappropriate development process/methodology	Sonchan and Ramingwong (2014) Sipayung and Sembiring (2015) Asif and Ahmed (2015) Sonchan and Ramingwong (2015) Reifer (2002) Hijazi et al. (2014b) Koopman (2011) Verner et al. (2008b) Murthi (2002) Asif et al. (2014) Liang et al. (2007) Wallace et al. (2004) Šmite (2006) Dey et al. (2007) Hu et al. (2013) Elzamly et al. (2016) Kaur et al. (2013) Koopman (2010) Verner et al. (2014)
R8DEV	Problems with new technology	
R9DEV	Inadequate infrastructure	
R10DEV	Unrealistic schedule	
R11DEV	Unrealistic resource planning	
R12DEV	Communication gaps	
R13DEV	Conflicts among team members	
R14DEV	Inefficient team capability	
R22DEV	Improper planning	
R23DEV	Project manager lacks experience	
R25DEV	Cultural diversity	
R26DEV	Lack of motivation	
R27DEV	Extensive personnel hiring	
R32DEV	Lack of top management commitment and support	

R35DEV	Unavailable customer contact	
R37DEV	Gold plating	
R38DEV	Developing the wrong software functions	
R40DEV	lack of project delivery milestones	
R42DEV	Backup issues	
R47DEV	Design is skipped or is created after code is written	
R50DEV	Inadequate management of change	
R54DEV	No update plan to the final software product	
R55DEV	No IP protection plan. No version control (IP)	
R56DEV	No backward compatibility and version management plan	
R60DEV	No training for managing outsourced relationships	
R64DEV	Project distribution	

3.6.1.3 ***Program Constraints class***

During a software project, the project could face external risk factors. According to Perera and Ranasinghe (2006)"These are factors that may be outside the control of the project but can still have major effects on its success or constitute sources of substantial risk". This class contains three elements. Resources is the first element, in which each factor that related to resources. The second element is the contract, as "Risks associated with the program contract are classified according to contract type, restrictions and dependencies". The third is program interfaces, as they contain some organisations or individuals outside the software project, such as subcontractors.

Table 3-5 Program Constraints risk factors

Code	Program Constraints risk factors	References
R15PR	Staff turnover	Sonchan and Ramingwong (2014) Verner et al. (2014) Sonchan and Ramingwong (2015), Asif and Ahmed (2015) Reifer (2002) Hijazi et al. (2014b) Koopman (2011) Islam et al. (2014) Lu and Yu (2012)
R16PR	Unrealistic budget	
R17PR	Resource insufficiency	
R18PR	User resistance	
R19PR	Lack of law enforcement	
R24PR	Government factors	
R30PR	Market demand obsolete	
R39PR	Subcontracting	
R41PR	Lack of staff experience	
R43PR	Natural disasters	
R51PR	Data privacy issues	
R59PR	Risk of problems with external tools and components	
R63PR	Inadequate knowledge/non-technical skills	

A. Product Engineering
1. Requirements
   a. Stability
   b. Completeness
   c. Clarity
   d. Validity
   e. Feasibility
   f. Precedent
   g. Scale
2. Design
   a. Functionality
   b. Difficulty
   c. Interfaces
   d. Performance
   e. Testability
   f. Hardware Constraints
   g. Non-Developmental Software
3. Code and Unit Test
   a. Feasibility
   b. Testing
   c. Coding/Implement-ation
4. Integration and Test
   a. Environment
   b. Product
   c. System
5. Engineering Specialties
   a. Maintainability
   b. Reliability
   c. Safety
   d. Security
   e. Human Factors
   f. Specifications

B. Development Environment
1. Development Process
   a. Formality
   b. Suitability
   c. Process Control
   d. Familiarity
   e. Product Control
2. Development System
   a. Capacity
   b. Suitability
   c. Usability
   d. Familiarity
   e. Reliability
   f. System Support
   g. Deliverability
3. Management Process
   a. Planning
   b. Project Organization
   c. Management Experience
   d. Program Interfaces
4. Management Methods
   a. Monitoring
   b. Personnel Management
   c. Quality Assurance
   d. Configuration Management
5. Work Environment
   a. Quality Attitude
   b. Cooperation
   c. Communication
   d. Morale

C. Program Constraints
1. Resources
   a. Schedule
   b. Staff
   c. Budget
   d. Facilities
2. Contract
   a. Type of Contract
   b. Restrictions
   c. Dependencies
3. Program Interfaces
   a. Customer
   b. Associate Contractors
   c. Subcontractors
   d. Prime Contractor
   e. Corporate Management
   f. Vendors
   g. Politics

Figure 3-2 Taxonomy of Software Development Risks

3.7 Summary

In this chapter, the most prominent concepts of risk have been reviewed, since the primary objective of this chapter was to identify, review and select the most relevant risk taxonomy, in addition to identifying the most important risk factors facing software projects.

Chapter 4: Success and software projects

Chapter 4: Success and software projects

4.1 Introduction

Success is the most important goal of any project. This chapter defines success and then investigates how success can be measured in projects by investigating the well-known criteria of success. The chapter concludes by identifying the most important success factors in a software project.

4.2 Success definition

One of the challenges that face any project management is how to define and manage success in projects. Moreover, everyone working on a project has their own view of a successful project. Success factors have been defined by Butler and Fitzgerald (1999)as the criteria in a project that have to be fulfilled and achieved so the project is concluded successfully.

Success can be defined as “The accomplishment of an aim or purpose” (Oxford Dictionary, 2016). Similarly, the Cambridge (Dictionary, 2016a) defines success as “the achieving of desired results, or someone or something that achieves positive results”. Based on these definitions, it can be stated that success has always has been linked to the achievement of a particular goal.

With regard to the success of a software project, this could be looked at from different perspectives, as success in a project can be identified using a variety of measures (Zhang et al., 2007). For example, a software project could be considered as successful if it delivered a product that achieved the planned goals and was within the budget, which are some of the metrics used to measure success (Pereira et al., 2008). Agarwal and Rathod (2006) noted that the success of a software project depends on delivering the product within the budget and within the planned schedule, and that the final product meets the agreed quality standard. But this research believes that the most suitable definition of a successful project can be found in , De Wit (1988)as he defined a successful project as follows: “The project is considered an overall success if the project meets the technical performance specification and/or mission to be performed and if there is a high level of satisfaction concerning the project outcome among key people in the parent organisation, key people in the project team and key users or clients of the project effort”.

4.3 Measuring success in projects

Software and IT project are the most risky projects as more of these projects fail or are not delivered on time compared to other projects, as has been mentioned. A joint study by McKinsey and Oxford University found that large software projects on average run 66% over budget and 33% over schedule (Chandrasekaran and Kaniyar, 2014). So the question of success in a software project depends on how we measure success. What are the most influential criteria by which to measure success?

Many researchers have tried to measure success in projects (Wateridge, 1995, Wateridge, 1998, Atkinson, 1999, Dyba, 2000). However, researchers are not the only ones attempting to do this; managers and organisations have also tried to find criteria to measure the success or a failure of a project, which appears to be a very difficult target (Thomas and Fernández, 2008). Furthermore, the identification of success differs depending on the stakeholders perspective of measurement (Berssaneti and Carvalho, 2014). According to Thomas and Fernández (2008), Wilson and Howcroft (2002) and Irani et al. (2001), although many works have been conducted with regard to measuring success there is no common agreed definition of success. But that does not mean there are no well-known criteria (for example, cost and quality) by which to measure success; it just means that there is still an argument about whether they are enough to measure success.

Of the most important things that should be carefully taken into consideration is whether it is possible to measure success in software projects. There are a number of criteria that are used in the measurement of software projects, and the most common are time, cost, scope and quality (Anda et al., 2009; Atkinson, 1999; El Emam and Koru, 2008; Kappelman et al., 2006; Lai, 1997; Sumner et al., 2006; Yeo, 2002). Above all, researchers have argued whether the information we have with regard to traditional software success criteria is enough, as in software projects it is almost impossible to apply only cost, time and quality and consider the project as successful. For example, De Bakker et al. (2010) studied the success of software projects between 1997 and 2009 and found that, even after meeting the traditional success criteria, some software projects still failed. The reasons behind that could be the complexity of software project, e.g., each stakeholder has their own success criteria and each one could be impacted by the interaction of factors that lead to success as well as by risk factors, as shown in Figure 4-1. Therefore, this research conducts a comprehensive study to determine the critical success criteria as well as the factors that lead to success in a software project.

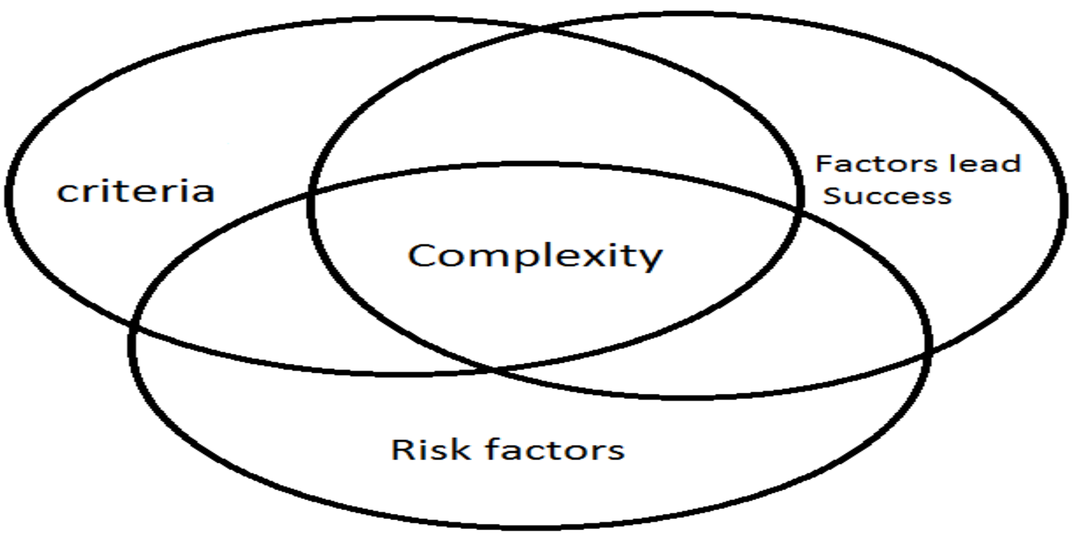


Figure **4-1** Criteria and Factors that lead to success

4.4 Success criteria and factors that lead to success

We note that general project management research has moved further than software development research in examining project success. Moreover, there are two different concepts that should be distinguished: the difference between success criteria and factors that lead to success. Project failure and success factors are understood to be elements that can be influenced to increase the likelihood of failure or success (Collins and Baccarini, 2004; de Wit, 1988; Müller and Turner, 2007; Pinto and Slevin, 1987). Project success criteria are used when making an evaluation of project failure or success (Collins and Baccarini, 2004; Müller and Turner, 2007). Consequently, success factors have a direct effect on success criteria; for example, ‘clear requirements and specifications’ is a factor of success that has an impact on success criteria like cost. Thus, although a factor does not represent success in a project in itself, it lead to the success of one or more success measurements.

This research will give an overview of the most agreed aspects of measuring success in projects, in which those criteria will be defined and explained from the viewpoints of both scholars and management teams as well as related stakeholders.

4.5 The Iron Triangle

Cost, time and quality have been over the years considered as fundamental criteria to measure success where, for example, Morris and Hough (1987) have built their framework on ways of measuring success over cost, schedule and quality. These concepts are some of the most used criteria to measure a project’s success or failure. Moreover, cost, schedule (time) and quality have been identified because of the importance of measuring the success of a project according to the ‘iron triangle’ or the ‘golden triangle’, which some professionals call the ‘Holy Trinity’ or the ‘triangle of virtue’ (Atkinson, 1999, Ika, 2009, Westerveld, 2003).

Even though these measures have been argued widely as the best to use, they are not the only criteria by which to measure success, but certain authors have stated that they are the "gold standard for measuring project success" (Barclay and Osei-Bryson, 2010; Meredith and Mantel, 2000; Pinto & Slevin, 1987; Papke-Shields et al., 2010; Berssaneti & Carvalho, 2014). In simple words, the iron triangle is considered successful if the project is delivered on time and roughly meets the planned budget, as well as meeting the requirements set by the stakeholders in addition to the customers' demands. Furthermore, ISACA (2008) states that "ensuring the delivery of project results within agreed-upon time frames, budget and quality by focusing on a defined program and project management approach that is applied to IT projects and enables stakeholder participation in and monitoring of project risks and progress".

According to Agarwal and Rathod (2006), the iron triangle has been used by professionals to measure performance in software projects; they also note that in some cases cost, time and quality have been used with other measures. This emphasises the iron triangle's importance as a measure of success for software projects, which have over the years been delivered over the agreed cost as well as not on time. Bloch et al. (2011) noted that budget and time are two of the biggest risks facing software projects in general; more than 45% of IT projects run over the budget and 56% will not meet the quality needed in the final product. The performance of software projects can also be measured by their ability to be delivered on time within the target cost and meeting the standards of product quality (Weber et al., 1994; Agarwal and Rathod, 2006).The iron triangle (cost, quality and time) is explained in more detail below:

4.5.1 *Quality in projects definition*

Tukel and Rom (2001) define project quality as "meeting customer's needs fully for the end product, reducing the reworking of non-conforming tasks, keeping customers informed of the progress of the project, and changing the course of work to meet the customer's emerging requirements".

4.5.2 *Quality in software projects*

Quality in software projects often may not be a priority for the project management and this is because a many managers give their entire focus during the project to delivering it as soon as possible. This is because a delay in the delivery of the product may cause it to lose its competitiveness in the market. From the other side, the delivery of a low-quality product or one that does not meet customers' needs may lead to the collapse of the project. This difficult equation makes quality difficult to understand worldwide in terms of software projects (Slaughter et al., 1998).

According to Rothenberger et al. (2010), software projects can be improved with respect to their quality. By using the proper scientific method, quality can be managed in software projects as they significantly share many of the same characteristics as engineering projects. One of the most important things that software project managers know is that they have to be prepared to raise the quality of a software product from the early stages of the project, for example finding programmers who are able to write high-quality code. The reason for the attention to quality in the earlier stages of the project is because often if a deficiency or weakness in the quality of the product is noticed it is considered to be too late to improve the quality as, in general, it is very difficult to revise the specifications to address issues relating to quality. (Reifer, 2006).

When looking for quality, there are important aspects that should be taken into account. The first is the technical side, which covers the technical characteristics. Second is the customer, as it is very important to find a product that meets the client's needs and meets his/her ambitions (Berssaneti & Carvalho 2014). From the perspective of the developers, the success of the product depends entirely on the extent to which the quality of the product meets the customer's needs.

4.5.3 *Cost*

Another important aspects of the iron triangle in determining the success of a project is the cost. Cost plays a key role in the success of technical projects. For example, Santos et al. (2013) noted that quality alone is not enough to measure IT project success; other very important aspects should be taken into account, such as the goal and the cost of the project. Linberg (1999) defined a successful software project as one that “meets its budget, delivery and business objectives”.

Weerasinghe et al. (2014) in their research on information systems projects identified three conditions that, if they occur in a project, can result in its failure. One of the conditions is when the cost of the project is greater than the financial return expected from it; this project is considered a failure. Financial returns from lucrative software projects as well as a lot of examples of software projects with huge financial returns led investors to invest in information technology projects. But the difficulty in measuring the cost of the project and the complexity that surrounds it is a big obstacle faced by most IT projects. Other studies regarding the reason for project failure show that inaccurate estimation is the root factor for failure in most software projects that fail (Jones, 2007; Jørgensen, 2005; Kemerer, 1987; Moløkken and Jørgensen, 2003).

Thomas and Fernández (2008) conducted a survey related to measuring success in IT projects in 27 companies. The results showed that cost of the project is considered one of the most important reasons for the success of an IT project. The majority of technical projects and software projects exceed the expected costs. The estimate cost of IT projects is significantly higher than what is expected for hardware, software and installation. One of the reasons for this is that the managers mainly estimate the value of the project at the planning stage. But in advanced stages of an IT project, such as testing and installation, other costs may have to be added, such as those relating to hardware and storage devices and the hiring of experts to solve any problems, for example, in related networking (Love et al., 2005). Furthermore, using the waterfall model usually increases the cost of the project generally, whilst using agile methodology has the ability to reduce the overall project cost due to its flexible nature. According to Dybå and Dingsøyr (2008), in some projects the choice of an agile methodology has reduced the project cost overrun by 25% . The size of the project could have a huge impact on the methodology used as this research, as has been mentioned, recognised that agile methodology is more suitable for small and medium projects, because it could add extra costs to a big project because of the difficulty of applying its manifestos and dividing the project in into mini development cycles .

In another study about the cost of IT projects in 2003 in the United Kingdom, it was found that 59% of IT projects did not comply with the expected cost of the project; more money had been spent on the project than had been agreed at the planning stage. Out of 421 projects investigated in the study, only 16% were delivered on time and for the proposed budget (Sauer and Cuthbertson, 2003). This result illustrates the importance of cost in the success and failure of technical projects in general, as it is clear that a failure to assess the budget plays a big role in the success of a project.

4.5.4 *Time*

The project completion time is also known as the time plan for the project, or project schedule. It is defined as the length of time necessary to complete the project, from the first phase of the project until the project is delivered to the beneficiaries. Time is one of the three parts of the iron triangle measuring the success of the project. It is considered one of the most important ingredients, and is measured by the success of the project. According to Phua (2004), a project is defined and its success is measured by whether it has been delivered at the agreed time, on budget and to the technical specifications.

Project time is also considered to be a very important element because the timing of the project has an effect on the success of other factors. For example, the project schedule also affects the cost of the project, as any delays in the delivery of the project increase its cost, which may in turn affect the profits expected from the project. It is one of the main reasons why projects fail. According to Sauer and Cuthbertson (2003), 45% of IT projects are not delivered on time. This figure reflects the large number of difficulties faced by IT and software projects with respect to time. Therefore, time in a project is considered to be one of the most important pillars by which project success can be measured. The software development method chosen plays an essential part in the software project's timescale. Agile methodology has been proven to save project time due to its continuous integration. Huo et al. (2004) noted that developers' time in debugging errors in the software will be saved if the agile methodology has been applied due to its continuous integration, which will in its turn save the project time and enable early detection of compatibility problems. On the other hand, the waterfall model has been recognised as being very time consuming due to its frozen stages (Balaji and Murugaiyan, 2012).

4.5.5 *Scope*

The goal of a project is the purpose for which it has been created, or the output of the project, which has been developed by the beneficiaries in order to provide a service. The scope is based on the view of those interested in the project's management as part of the quality. However, some authors state that quality serves as the scope. McLeod and MacDonell (2011) noted that the iron triangle of project success found help the project to rich it is scope in software projects. This may be a perception that is the closest to reality. Software project was completed within budget and with good quality and in the agreed time but it did not meet with the project scope it led to define the project as non-successful project.

Burke (2013) stated that "effective scope management is one of the key factors determining project success". Scope is one of the important aspects used in measuring success in software projects. In research about the areas by which to measure success in software projects, Agarwal and Rathod (2006) noted that, in the professionals' view, the most important aspect is the scope of the project. This results shows the importance of scope in software projects, and puts more emphasis and focus on this area of measuring success. In addition to that, the software methodology helps the project stay within the specified scope. In the agile methodology, customers are asked to be involved in the project's overall scope. For example, in XP's weekly meeting the customers are asked to choose and specify the weekly development cycle scope, which the development team implement in order to achieve the project aim (Lee and Xia, 2010). Moreover, in the waterfall model the scope is defined and specified and all the requirements needed to reach the project goal should be well documented in the requirement stage, which helps the project to be successful and stay within the specified scope.

4.6 Stakeholders and success criteria

Researchers have noted that, when considering success, it is important to identify the stakeholders (Munns and Bjeirmi, 1996). Based on the above, it is very important to determine who are the stakeholders related to a project, because doing so will enable the researcher to uncover their views on the complexity of the success factors and criteria and risk factors in related to the project. So the question is which stakeholders' view of success should be taken into account?

According to Davis (2014), project managers are the most highly cited stakeholders when measuring project success. Likewise, the success factors found by this study are more management in nature rather than being technical. Davis (2014) also found that the management team and the development team are two of the most important stakeholders. First is the management team, which covers the people related to management – “Board, director, executive, executive management, investor, project executive, portfolio director, programme director, owner, senior management, sponsor, top management, project sponsor”(Davis, 2014). Second is the development team or “Project core team”; the stakeholders in this category are related to the team, such as “Engineers, team leader, project leader project manager, project personnel, project team leader, project team and team members as well as other organisational involvement (e.g. business departments)”. Although these stakeholders are different, they do share some views regarding success criteria and factors.

4.7 Critical factors leading to success and taxonomy

In this research, 37 success factors of software projects have been identified, based on the analysis of our extensive literature search.

These success factors were identified in many publications. Some factors used in this study have been reviewed in the publication of Nasir and Sahibuddin (2011b). In their research about the critical success factors for software projects, which was conducted via a comparative study, they identified 26 critical success factors related to software project success in all the articles they investigated from 1990 to 2010. Furthermore, they found that non-technical factors were 94% while the technical factors identified were just 6%. This research correspondingly has a similar result where the majority of the critical success factors identified were non-technical factors. A study by Khan et al. (2011) identified 22 success factors related to offshore software development outsourcing (OSDO) projects, which are projects developing high-quality software in countries where wages and costs are low.

As mentioned earlier, the success factors are different from success criteria. Success factors are factors related to a project directly or indirectly besides affecting the project in terms of increasing the chances of success.. On the other hand, success criteria are assessment results criteria by which a project could be evaluated in terms of its success or failure once it has been completed. These success criteria are affected positively when the success factors are applied and the risk factors avoided in the project. Additionally, success could not be achieved when the success factors are not applied and there is no risk management in the project. Success factors are generally composed into two categories in software projects. The first category is the non-technical success factors. These are success factors such as management and the methods used in project management, as well as people involved directly or indirectly in the project, such as managers and users. Another non-technical aspect is the project environment and its impact on the success or failure of the project as well as the general environment of the organisation. Non-technical success factors associated with the project are considered to be very important in terms of their impact on the project, which confirms the conclusion reachedby (Khan et al., 2011).

The second category is the technical success factors; these are technological factors which, if taken into account, affect the success of project but this effect remains very low compared to other factors categories.

This study also confirms what has been mentioned before by Khan et al. (2011), (Bayona-Oré et al., 2014) as they noted that the non-technical factors are important to the success of the project. Furthermore they found that approximately 96% of the success factors are non-technical factors, which shows that only 4% of them are technical factors. In conclusion, non-technical factors contribute significantly to the success or failure of software projects.

With regard to the taxonomy, this research has used SEI taxonomy, which is same taxonomy used for the risk factors, as the software projects have already been classified into three classes. Also, having the same classes helps to investigate, uncover and distinguish the relationship between the success and risk factors. In addition, it enables us to identify if success factors will only affect factors in the same class or will also have a significant effect on risk factors in other classes. Success factors are classified into three classes, as follows:

4.7.1 *Product Engineering class*

Table 4-1 Product Engineering success factors

Code	Product engineering success factors	References
S1ENG	Clear requirements and specifications	Nasir and Sahibuddin (2011b) Darwish and Rizk (2015) Devedzic (2007) Reel (1999) Purna Sudhakar (2012)
S2ENG	Clear objectives and goals	
S10ENG	Familiarity with technology/development methodology	
S22ENG	Managing the complexity of project size, number of organisations involved	
S34ENG	Getting code from a high-quality, reliable and stable community	
S35ENG	Extensive testing for quality and careful selection of code	

4.7.2 *Development Environment class*

Table 4-2 Development Environment success factors

Code	Development Environment success factors	References
S3DEV	Realistic schedule	Nasir and Sahibuddin (2011b) Ikonen and Abrahamsson (2013) Iqbal et al. (2011)
S4DEV	Efficient project management	
S5DEV	Top-level management support	

S7DEV	Effective communication and feedback	Ahimbisibwe et al. (2015) Purna Sudhakar (2012) Imtiaz et al. (2013) Keil et al. (2003) Basri and O'Connor (2011) Bhoola (2015) Juárez-Ramírez et al. (2013) Keil et al. (2013) Nienaber et al. (2012) Sheffield and Lemétayer (2013)
S9DEV	Skilled and sufficient staff	
S11DEV	Appropriate development processes/methodologies	
S12DEV	Proper planning	
S13DEV	Up-to-date progress reporting	
S14DEV	Effective monitoring and control	
S16DEV	Good leadership	
S17DEV	Risk management	
S18DEV	Change management	
S20DEV	Committed and motivated team	
S21DEV	Good quality management	
S24DEV	Clear assignment of roles and responsibilities	
S25DEV	Team environment	
S26DEV	Customer training and education	
S29DEV	Team capability	
S31DEV	Organisational culture	
S32DEV	Stability of organisational environment	
S33DEV	Project criticality	
S37DEV	Team training	
S38DEV	Project manager dedicated to the project	

4.7.3 *Program Constraints class*

Table 4-3 Program Constraints success factors

Code	Program Constraints success factors	References
S6PR	User/client involvement	Nasir and Sahibuddin (2011b) Ahimbisibwe et al. (2015) Purna Sudhakar (2012) McLeod and MacDonell (2011) Bayona-Oré et al. (2014) Khan and Khan (2013)
S8PR	Realistic budget	
S15PR	Adequate resources	
S19PR	Appropriate infrastructure	
S23PR	Pilot project performance	
S27PR	Efficient contract management	
S28PR	Good performance by vendors/contractors/consultants	
S30PR	Political stability	
S36PR	Commitment of stakeholders	

4.8 Summary

This chapter has defined success as well as reviewing some of the concepts related to it. Moreover, this chapter has identified and reviewed the criteria that are the measures of success, in addition to identifying the most important factors that help software projects to succeed.

Chapter 5: Complexity and network analyses

Chapter 5: Complexity and network analyses

5.1 Introduction

Complexity is one of the more interesting topics in the study of software projects. This chapter defines complexity, and reviews its most important characteristics. In addition, the researcher reviews the relationship between complexity and the failure of software projects, and ends by shedding light on the definition, concept and characteristics of networks.

5.2 Complexity definition

According to the Oxford Dictionary (2016b), complexity can be defined as “The state or quality of being intricate or complicated” and “A factor involved in a complicated process or situation”. Furthermore, the Cambridge Dictionary (2016a) defines complexity as “the state of having many parts and being difficult to understand or find an answer to”. It can be seen that the Oxford Dictionary defines complexity as a situation in which the components are not total understandable. Although the Cambridge Dictionary (2016a) has a similar definition, its definition mentions that, in order to have complexity, two or more than two factors have to be involved where either the outcome or the process are not understandable. Wernham (1985) has similarly defined complexity, adding that complexity occurs because of the interaction of multiple interrelated components. Thus, based on this, there are two concepts related to complexity (understanding and interaction) in a situation. Complexity in relation to software projects has also been defined. The Lagerstrom et al. (2014) definition is similar to those previously mentioned: “degree to which a system or component has a design or implementation that is difficult to understand and verify”.

Above all, the concept of the complexity is widely debated, as there are no agreed concepts about it, and it has become important to show that there are differences between the several aspects related to complexity, such as the difference between complexity and complicated. In most cases, complexity is about not fully understanding a phenomenon or an interaction between components in an environment, whilst complicated has the opposite meaning to simple: there are several components in a system but this system is not understandable.

In addition to the above, it is important to distinguish that the terms ‘complicated and ‘complexity’ are different. Cilliers and Spurrett (1999) have explained the differences between the two terms as “it is useful to distinguish between the notions ‘complex’ and ‘complicated’. If a system despite the fact that it may consist of a huge number of components—can be given a complete description in terms of its individual constituents, such a system is merely complicated. Things like jumbo jets or computers are complicated. In a complex system, on the other hand, the interaction among constituents of the system, and the interaction between the system and its environment, are of such a nature that the system as a whole cannot be fully understood simply by analysing its components. Moreover, these relationships are not fixed, but shift and change, often as a result of self-organisation. This can result in novel features, usually referred to in terms of emergent properties”. From that it is also notable that the words ‘fully understood’ are attached to the term ‘complicated’, whilst ambiguity and the inability to fully understand a system are attached to the word ‘complexity’, as has been mentioned.

The concept of complexity of interaction is referred to as interfaces between the elements or factors in a specific environment. The importance of a factor in terms of its interaction relies on several characteristics. One of the most important is the location of the factor and how it influences other factors in the system or project. Furthermore, these interactions could be between systems, locations and humans (Fitsilis, 2009).

Other important aspects of complexity are its science and theory. Couture and Valcartier (2007) state that scientists and researchers have discovered that not all systems are understandable or linear, and define complexity theory as: “These non-linear systems are showing behaviours and types of orders that are hard to predict, even when they are governed by simple rules.”. Similarly, Johnson (2009) has defined complexity science as: “the study of phenomena which emerge from a collection of interacting objects”. Based on that, it is important to understand complex systems in more detail, as follows.

5.3 Complex systems

A complex system is two or more components, factors or parts that interact with each other and work to provide a service or a function (Johnson, 2009). These parts may be divided into subsystems. These subsystems may in turn also be subdivided into other subsystems and so on. For example, the galaxy contains several solar systems. These solar systems are divided into many planets, which contain many species and so on. Simon (1991)defined a complex system as: “one made up of a large number of parts that interact in a non-simple way. In such systems, the whole is more than the sum of the parts, not in an ultimate, metaphysical sense, but in the important pragmatic sense that, given the properties of the parts and the laws of their interaction”.

These definitions show that subsystems are an important element in a complex system, as complexity and complex systems are defined by their subsystems, which could be ambiguous in nature. Furthermore, ambiguity plays an important role in determining a complex system. Each subsystem could have certain laws inside it but contain interacting that, as a whole, are difficult to understand or predict. In addition, complex systems do not have a centralised control, as they are non-linear (Vasileiadou and Safarzyńska, 2010). This supports the claim that these systems are difficult to understand or manage. The word non-linear also shows the difficulties of predicting the behaviour of the system. Moreover, a complex system has many characteristics, which will be discussed in more detail later on.

5.4 Characteristics of complexity

According to Saurin et al. (2013), there is no agreed definition for the concept of complexity, as has been mentioned. However, each system has its own characteristics that distinguish it from other systems, and these characteristics may be important in determining whether a system is complex or not, as a complex system's characteristics could be used to define complexity (Johnson, 2009). Through an investigation of previous literature this research has identified the characteristics of complex systems as follows:

5.4.1 *Non-linearity*

According to Parker and Stacey (1994), a linear system can be defined as a system in which results can be anticipated. In many cases, there is one reaction for a single action. From this definition, it can be deduced that a non-linear system is one that may contain more than one reaction because of a single event. Because of these many results, outputs or reactions, it is hard to understand the system's behaviour, which agrees with what Bellavite (2003) notes about the reason behind unpredictability of non-linear systems: it could be because the relationship between output and input is non-logical and may even be considered as "chaotic phenomena". Furthermore, the reason behind the non-logical behaviour of the components in a non-linear system is that elements in the system are working in different environments, under different rules and at different times (Fontana, 2010). As result, non-linearity is one of the most important characteristics in defining complexity and the complex system.

5.4.2 *Self-organisation/(emergence)/resilience*

Self-organisation in complex systems is the ability to create an order between the elements, parts or factors of a complex system, from a disordered situation. Such order occurs without any external interference. That might euphemistically be called organised chaos. This arrangement does not always mean that the system organised itself if it contains many elements.

There may even be a complex system that does not contain many elements. But complexity is shaped because of the number of interactions between these elements. In general, these interactions are self-organised, and it is difficult to understand how they have organised themselves (Bellavite, 2003). Fontana (2010) has noted that the main advantage of having self-organisation in a complex system is to create a balance in it. Parker and Stacey (1994) explain this characteristic as the transformation from the state of organisation to a state of organised chaos through an incomprehensible internal order. This definition does not contravene those previously stated; it looks at self-organisation from a different perspective but with the same concept. This is because both explanations talk about the existence of self-organisation in the concept of a chaotic and disordered system.

5.4.3 *Negative and positive feedback/history*

Feedback is the process of influence among the elements. In other words, the change that occurs in an element in the system affects another element or more in the system. There are two types of feedback. First, positive feedback: this, for example, is when the increase occurring in an element causes other elements to also increase. Which the impact of these elements in it is tern it could impact on the original element or source of change, which will create a loop of change inside the complex system (Vasileiadou and Safarzyńska, 2010). One example of positive feedback is the temperature of the Earth's atmosphere. The increase in atmospheric temperature increases the water temperature of seas and oceans, which leads to increased evaporation. This vapour rises into the atmosphere again and causes global warming, and then increases the heat of the Earth's atmosphere. In contrast, negative feedback occurs when an increase in an element leads to a decrease in another element or other items, which in turn has an adverse result in decreasing the original element.

Feedback in a complex system could be used as a tool to increase the resilience of the system (Saurin et al., 2013). Kay (2004) noted that there are two types of change in in relation to feedback loops in a complex system. If the environment changes, the feedback loops usually maintain the same patterns, but if the system changes the feedback loops tend to change rapidly; when this occurs it is extremely difficult to predict the outcome or the change that will happen in the system.

5.4.4 *Wide diversity of elements*

Diversity in complex systems is the presence of several elements that fall under multiple categories, which in turn may create complexity. These items may fall under multiple categories such as “levels, division of tasks, specialisations, inputs and outputs” or “degree of co-operation, degree of shared objectives and degree of information exchange” (Saurin et al., 2013). This shows that this diversity may be in the categories of items or in the type of interaction. Furthermore, Cilliers and Spurrett (1999)noted that there are two characteristics of diversity in complex systems: a large number of elements and the interaction between these elements. Usually, a complex system contains a large number of elements and these elements are distributed into many groups (Gatrell, 2005). The second characteristic is diversity of interactions between these elements. This diversity is found either at several levels of interactions or in the method or interaction (Heylighen). One of the most important benefits of diversity in a complex system is its ability to fill a vacuum in the system if an element is missing by reorganising itself to fill the gap (Fontana, 2010).

5.4.5 *Emergence*

Emergence is one of the most important characteristics of complex systems. It appears when the system elements interact with each other so that the output, attributes or behaviour of the system as a whole is new and unpredictable. One of the main reasons for emergence is the unpredictable and self-organising behaviour of complex systems (Gatrell, 2005, Vasileiadou and Safarzyńska, 2010). There are two types of emergence. The first is the emergence of a new output, attributes or behaviour in the whole system. The second is the emergence of output, attributes or behaviour that already exist in the system but which were not visible as a result of the interactions between the elements or the absence of an element (Mikulecky, 2001). From this it can be concluded that emergence in complex systems is the behaviour, attributes or outcomes of the system level being contrary to the behaviour, attributes or outcomes of the element level as a result of interactions, unpredictable and self-organising system.

5.5 Project complexity

According to the Cambridge Dictionary (2016a), a project can be defined as “a piece of planned work or an activity that is finished over a period of time and intended to achieve a particular purpose”. Also, Vidal et al. (2007) explained that the system is a framework composed of several parts aimed at reaching a certain goal or providing service in a given time period. This research has adopted Vidal and Marle (2008) definition of project complexity as: “Project complexity is the property of a project which makes it difficult to understand, foresee and keep under control its overall behaviour, even when given reasonably complete information”. This definition agrees with what has been mentioned about interaction and ambiguity, two of the main attributes of complexity in general.

According to Frame (2002), complexities cannot be separated from projects because projects are always complex and not fully understood. Complexity is considered very important in the field of projects and project management (Cicmil et al., 2006). The reason for this is that most projects contain a lot of overlapping parts. Complexity Science is interested in studying the interactions that occur between these parts (Lu et al., 2015). Therefore, the existence of these attributes in projects led to the existence of project complexity. Furthermore, when these parts of the project interact with each other, it affects the general characteristics of the project (Xia and Chan, 2012). In addition, the study of complexity has led to more understanding of projects, which in turn, even if did not provide a full understanding of project complexity, gives decision-makers the capacity to make the right decision for the success of a project (Lu et al., 2015). Moreover, the study of project complexity has uncovered the interactions that occur between risk factors and project stakeholders (Ackermann et al., 2014). Knowledge of the complexities of a project helps managers in selecting the appropriate method for project management. According to Hill (1991), the ability to manage the complexity facing a project is the most important characteristic that project managers should have. Also, Baccarini (1996) noted that six points show the importance of uncovering complexity in a project, as can be seen in the table below:

Table 5-1 Importance of complexity to the project management source: (Baccarini, 1996)

Project complexity helps determine planning, coordination and control requirements
Project complexity hinders the clear identification of goals and objectives of major projects
Complexity is an important criteria in the selection of an appropriate project organizational form
Project complexity influences the selection of project inputs, e.g. the expertise and experience requirements of management personnel
Complexity is frequently used as a criteria in the selection of a suitable project procurement arrangement.
Complexity affects the project objectives of time, cost and quality. Broadly, the higher the project complexity the greater the time and cost.

Projects contain many characteristics and interdependence is one of the most important. Moreover, interdependences are the relationships between the elements of a project and how they impact on each other. In relation to interdependences in a project, Rodrigues and Bowers (1996) noted that “experience suggests that the interrelationships between the project’s components are more complex than is suggested by the traditional work breakdown structure of project network”. From their explanation, this research concludes two important points. First, the interdependence between the parts of the project is not understood in general. This is could be due to the impact of the complexities of the project. The methods used to analyse and identify the interdependences of a project are ineffective and inadequate. Building a network of risk interaction would help to understand the complexity of a project (Fang et al., 2012), and centrality is a major factor in understanding which factors are interacting in a project environment (Stacey, 2007).

5.5.1 *Complexity of software project failures*

Software projects are one of the most commercial areas in which investment has been made during recent decades. However, they are one of the riskiest and less successful types of project. For example, Cerpa and Verner (2009) noted that the success rate for software projects has not gone up significantly compared to the amount of research that has worked to raise the success rate. Where it was observed that during previous years software projects were not completed on time, within the planned budget or in accordance with the specified quality. At the beginning of the year 2000 software project failures cost the US economy nearly $75 billion (Charette, 2005a). A Standish Group study of 2009 on software projects divided projects into three categories: successful, failed and challenged – which are projects that have experienced partial failure with regard to not being completed on time, over budget, and not to specified scope and/or quality. The report found that 66% of software projects fail either in whole or in part, while only 34% of projects are delivered according to the criteria that have been previously identified and considered successful (Hussain et al., 2016, Kamuni, 2015). This is an increase in the success of software projects compared to the group's 1994 report where the success rate was only 16% (Wiklund and Pucciarelli, 2009). Although the increase had doubled, the number of failed projects is significantly high. In the 2015 Standish Group study report, they studied nearly 50,000 IT projects and found that 29% of the projects had been successful, 19% had failed completely or been cancelled, and 52% had not been delivered as specified by the project criteria (time, quality, cost or scope) (Hastie and Wojewoda, 2015b). These statistics support Cerpa and Verner (2009) claim that software project success has not increased compared to the amount of research done to increase the success rate, as has been stated earlier. Also, it shows that there is still much work to be done in order to raise those numbers.

Cerpa et al. (2016) stated that have been a noticeable number of studies on software project failure from the user and the customer's perspective. However, they have not considered the development team’s perspective significantly. The development team plays one of the most important roles in the failure and success of IT projects, as many of the problems that lead to the failure of a software project can be understood and strategies developed to help to resolve them through the development team. In addition, the authors also stated that some improper practices by the management team also contribute to increase these percentages. Managers’ experience is very important in reducing the complexity of a project, because it helps them to predict the occurrence of a risk event during the project (Li et al., 2008).

Complexity is one of the most important topics that are studied in terms of projects in general and software projects in particular. That is due to its important and significant role in system development (Xia and Lee, 2005). Complexity might be considered one of the most important reasons for the failure of a huge number of software projects. Nguyen (2014) and Calleum (2014) noted that nearly 35% of software projects fail because of inability to understand their complexity. The difficulty could be because, when the team realises that they have underestimated the project’s complexity, it will often be too late to take steps to ensure the project’s success (Jørgensen, 2014). Another reason could be because it is difficult to study the complexity in all phases during a project (Nguyen, 2014). Furthermore, according to Petkova and Petkov (2015), “understanding software project complexity is an essential precondition for better IT project management”.

There are a number of reasons that increase the complexity of a project and thus increase the risk of its failure. The emergence of new technologies results in the existence of certain system structures that cause the emergence of unexpected risks and increase the project’s complexity (Sonchan and Ramingwong, 2014).

Complexity has been always associated closely with project success criteria. But Lehtinen et al. (2014) stated that complexity arises because of the interaction between time, cost and quality. Furthermore, complexity occurs between risk and success factors associated with these criteria. In addition, it is noted that the greater the size of the project the more complex it is (Li et al., 2008).

According to Fitsilis (2009), researchers must carry out more studies on the complexities of software projects in order to understand this topic as the current studies do not provide enough understanding about the phenomenon. He added that, in order to understand the complexities, two important points need to be considered, the first is that in order to understand the complexities we must study the entire software project; and the second is to understand the characteristics of the interactions between the system components or factors. This highlights the importance of interaction and interdependency between the elements of a software project, where the interaction between the project's factors or components can be measured by using network analysis. According to Petkova and Petkov (2015), network analysis helps to determine the importance of a factor in the software project and how it influences the overall project. These results help us to understand the project complexity.

The literature review illustrates the importance of complexity for software projects as follows. First, in order to study the complexity, we must study the project as a whole. Second, complexity has been associated closely with success criteria, as mentioned previously. Third, software projects contain risks and factors of success. All of these factors and criteria interact in a software project. Most studies have focused on one of these aspects without taking into account all the other aspects. Some studies (Na et al., 2007, Tiwana and Keil, 2004, Ruan, 2010, Li et al., 2008, Sonchan and Ramingwong, 2014, Hu et al., 2013) focused on the risk in software projects without taking into consideration the success factors, while other researchers have focused on the management and success factors in software projects and how they will help in raising the success rate, without paying much attention to the complexity of the interaction of those factors with other components in the project (Purna Sudhakar, 2012, Reel, 1999, Darwish and Rizk, 2015, Chow and Cao, 2008, Ahimbisibwe et al., 2015, Nasir and Sahibuddin, 2011a). Furthermore, related to the study of complexity in software projects, there have been some attempts to understand complexity, such as in Fitsilis (2009), where he tried to come up with a model to help to understand the complexity in software projects. This research believes that one limitation to his work is the number of factors involved. Also, the success factors have not been included. In addition, the interaction between the factors was not studied. Fitsilis (2009) has also concluded that more work has to be done in order to understand complexity, and he mentioned interaction as one aspect. Petkova and Petkov (2015) proposed a model of factors affecting software project complexity, but this research argues that neither the interaction nor the success factors and criteria have been included in the model. Therefore, this research found that there is a gap in the study of interactions between success and the risk factors with success criteria in software project interaction. The study of these factors helps us to understand the mechanisms through which the elements interact and cause complexity. Understanding these interactions could lead to the building of strategies and plans to deal with and predict these events, which in turn could help to increase the chances of software project success.

5.6 Network analysis

5.6.1 *Network definition*

According to the Cambridge Dictionary (2016a), a network can be defined as “a large system consisting of many similar parts that are connected together to allow movement or communication between or along the parts, or between the parts and a control centre”. The Oxford Dictionary (2016b) defines the network as “A group or system of interconnected people or things”. From these two definitions, it is clear that a network contains parts, factors or components. In addition, both definitions include the word communication or interconnected. This shows the importance of interaction as a main characteristic of a network. This research also notes that there are many similarities between the definition of a network and the definition of a system, as previously stated. Based on that, a network, in simple terms, is a system containing nodes (parts, factors, people or things) linked to or interacting with each other. Furthermore, this focus on interaction is known as network thinking, which, according to Mitchell (2009), is the tool that has been used to investigate and understanding complexity from the interaction point of view for the last five decades. Interactions are described as the heart and soul of complexity According to Fang and Marle (2013), “The complexity of [a] project leads to the existence of a network of interdependent risks”. Based on that, this research will use network thinking to uncover the complexity in the software development project.

5.6.2 *Network science*

Scientific development in recent decades has led to the emergence of new methods that enable researchers to explore networks more clearly. This in turn led to the existence of agreed concepts in the study of these networks, which is known as network science. Network science is not just about the study of large networks; it also helps to understand the complexities that appear in these networks (Estrada et al., 2010). Furthermore, according to Baggio et al. (2010), one of the most important methods used in order to understand the phenomenon is through the study of the interactions, communication and links between the elements of this phenomenon. The science behind the uncovering and study of complexities in this interaction is network science (Baggio et al., 2010). Accordingly, the networks can be considered as a mirror for the interactions that occur between complex system elements. Therefore, this research will use network science to explore and uncover the interactions that occur between elements in the software development project by applying network theories and graph theory and characteristics. Through network analysis, this research aims to identify the most important factors via their centrality in the network.

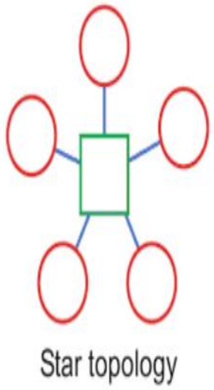
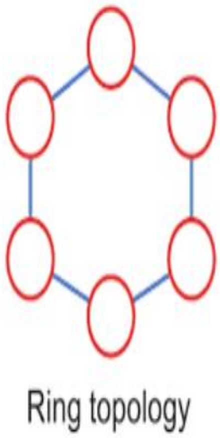
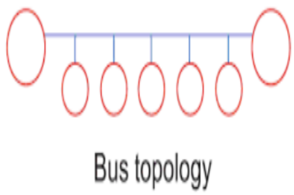
5.6.3 *Network analysis*

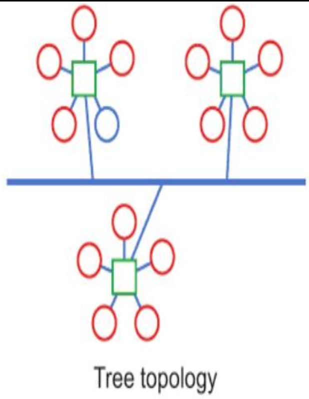
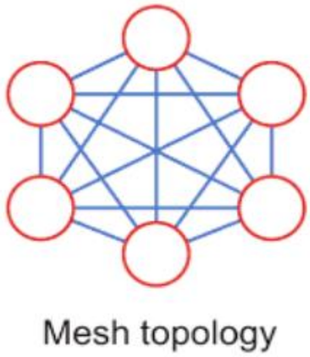
Network analysis is used to understand the complex system through dividing it into parts, components, factors or things, and then illustrating the interactions between these parts. The researchers found that the networks can explain complex systems because most networks share a large number of properties (Daim et al., 2016). According to Baggio et al. (2010), "Network research has revealed that network behaviours and processes can be explained based upon the properties of a system's general connectivity and studies have found that the topology of many complex systems has been shown to share Fundamental properties". Furthermore, complexity can be understood via the network analysis and visualisation (Durland and Fredericks, 2005, McSweeney et al., 2014). This research will now cover three aspects of network analysis. First, it will cover the main topologies. Second it will cover the network centrality measures. Third, it will cover network characteristics.

5.6.4 *Network topologies*

There are many types of network, each with its own shape. But those shapes can be classified into many topologies, as the network topology can be determined based on the interaction that happens between its parts. The elements of the network are called nodes and/or vertices and interaction between them can also be called edges. The table below explains the most common topologies:

Table 5-2 Network topologies

Topology name	**Explanation**	**All shapes have been taken from Herbert and Dyer (2005)**
Star topology	In this topology, a central node is connected to all other nodes in the network. Note that the other nodes do not directly interact with each other; they interact through the central node.	
Ring topology	In this topology, all nodes are connected with each other as a circle. Thus, each node interacts with only two other nodes in the network, and each node has the same centrality value in the network.	
Bus topology	Similar to the ring topology, in this topology all nodes interact with two other nodes, except for the first and last nodes, as they only have one interaction. Thus, not all nodes have the same centrality value.	

Tree topology	In this topology, shapes of the start and bus topologies are combined. There are some or many star rings interacting with each other via a bus topology.	
Mesh topology	In this topology, all the nodes are connected to each other, and thus all factors have the same centrality importance to the network.	

5.6.5 *Centrality measures*

Centrality measures are mathematical measures used in network analysis in order to understand the role of a node or nodes in a network, which in turn determines the importance of a node to other nodes in the network. According to Daim et al. (2016 and Freeman (1978), the centrality of a node is determined by its location in the network. Centrality measures help in understanding the interactions that occur between all the nodes in the network (Akhtar, 2014). Many researchers use four measures to measure centrality: “betweenness, closeness, degree, and eigenvector centrality” (Daim et al., 2016, Akhtar, 2014, Freeman, 1978, Croci and Grassi, 2014). This research will use these four measures in the study of complexity in a network. The following provides an explanation of these measures:

Betweenness centrality: Centrality is determined by calculating the number of times that a node appears when counting the shortest paths in the network. A node is said to have high centrality if the betweenness value is high compared to the rest of the nodes. The importance of this measure is due to the fact that the node with the highest betweenness controls the interaction among the rest of the network nodes. The location of the node is important in this measure Because it affects the number of indirect interactions that occur among the nodes (Freeman, 1978).

Degree centrality: Degree centrality is determined in a network by counting the number of direct interactions between the node and other nodes in the network. The significance of the central class is to illustrate the direct influence and impact of a node on the other nodes in the network. Also, the degree centrality could be an indicator of the activity level of a node in the network.

Closeness centrality: Closeness centrality is similar to degree centrality, but it calculates the average number of direct and indirect interactions between the node and other nodes in the network (McSweeney et al., 2014), while the degree centrality is concerned only with the direct interactions. The importance of closeness is that it shows how close the node is to the rest of the nodes in the network; this in turn explains the speed influence and impact of a node on all nodes in the network(Brandes, 2001). The node with the lowest closeness value in a network is the most central node in the network.

Eigenvector centrality: This measurement depends on the node's neighbours more than o the node itself in determining its centrality. Here, the node is considered as central not by its location, shortest paths or the number of direct interactions but by the number of central nodes to which it is connected (McSweeney et al., 2014, Bonacich, 1972) . In other words, a node is considered to have high eigenvector centrality if it has the largest number of connections to central nodes although it could have low degree centrality.

To sum up, the concept of centrality addresses the issue of how strategic the position of an entity within the network is. Usually, centrality is measured as a property of a single node within the network to evaluate the 'reachability' or the 'importance' of this node. The table below provides more explanation of the four centrality measures in terms of their meaning in network and risk.

Table 5-3 Network centrality measures

Centrality	**Meaning in network**	**Type of centrality**	**Importance**
Degree	Number of edges directly connected to other risk and success factors	Local centrality measure	Direct interaction between factors
Closeness	How close a factor is to every other factor in the network	Local centrality measure	Efficiency and ability of reach 'impact' other factor and independence within the network
Betweenness	Measure of the ability of a factor to control the flow of communication in a network's 'shortest paths'	Global centrality measure	Measures direct and indirect connections between two factors as well as analyses connections with the three factors involved
Eigenvector	General view of the centrality of a factor by showing how many central factors it is connected to.	Global centrality measure	Factor centrality is not determined by the number of direct connections but by the importance of the factors connected to.

5.6.6 *Network centrality characteristics*

There are several tools used to understand and uncover the general characteristics of a network. One of the most important studies that has used these tools to study complexity was conducted by Boussabaine and Vakili-Ardebili (2010). In their research to understand complexity of design in buildings they used a survey as the data collection tool. Accordingly, they built a network that described the interactions that occur between elements in the network, so that the nodes in the network represented the factors and edges represented the interaction. Furthermore, they used some statistical functions in determining the characteristics of the network from three centrality measures (betweenness, closeness and degree) points of view. Some of the functions used in their research are also used by this research into the study of the general characteristics of complexity of risk and success factors & criteria networks; these are Minimum, Minimum, Maximum, Sum and Mean. Minimum is used in order to determine at least a centrality value in every measure except closeness where the lowest number represents the most central factor. Maximum is used in order to show the highest centrality value in the betweenness and degree centrality. Sum is used in order to obtain an overall value for each measurement in the network. Mean is used to illustrate the general characteristics of each measurement in each network.

5.7 Summary

This chapter has defined complexity and has also clarified what a complex system is and what its characteristics are. It has also reviewed the relationship between complexity and the failure of software projects, and the concepts of network analysis.

Chapter 6: Methodology

Chapter 6: Methodology

6.1 Introduction

In this chapter, the research gives an overview of the available methodologies used in scientific research. Also, this research outlines the qualitative, quantitative and mixed methodologies, and explains the main reasons behind using the mixed method. Furthermore, the research process, data collection and data analyses are also explained in detail.

6.2 Research Paradigm

A research paradigm is a “philosophical framework that guides how scientific research should be conducted” (Collis and Hussey, 2013), where the research paradigm or research philosophy contains logical, scientific concepts and stages which help to reach the research aim. One well-known analogy on how research methodology is conducted is onion stages, as can be seen in Figure 6-1. The onion stages help the researcher understand the research methodology and how to determine the appropriate way to answer the research questions (Saunders et al., 2012). The onion presents several stages and it is important to understand each stage before moving to the next one. Furthermore, the research philosophy presents an outline of the onion, which means it is important to understand this layer before moving to the next layer. Research philosophy is all about how a researcher views the world (Creswell, 2009). Research philosophy is a logical structure of ontological and epistemological beliefs (Guba and Lincoln, 1994).

Ontology is about realism and the study of the nature of reality (Healy and Perry, 2000). Saunders et al. (2009) argued that objectivism and subjectivism are two aspects which are always linked with ontology. Additionally, both aspects are able to produce knowledge in research. Subjectivism considers social actors’ perceptions and consequent actions, in relation to their reality and existence. On the other hand, objectivism looks at social entities which exist in reality outside the social actors.

Epistemology is the methodology or technique that is used in research and shows the view of the researcher to reality (Healy and Perry, 2000). Eriksson and Kovalainen (2015) argue that the epistemological approach or methodological approach help researchers with how to choose the appropriate method of data collection: qualitatively, quantitatively or both.

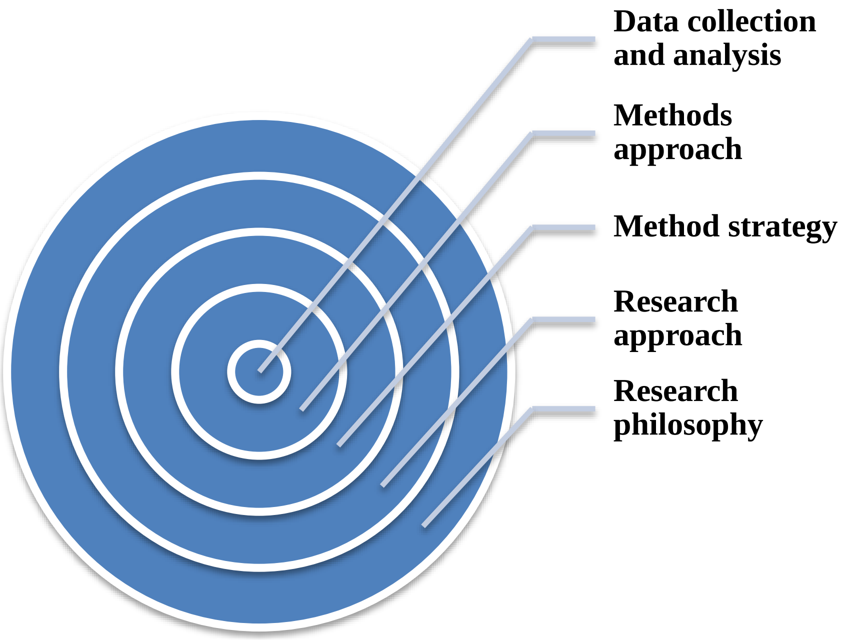


Figure 6-1 Research onion stages source: Nesensohn (2014)

6.3 Research approach

In scientific research, usually the search for knowledge depends on how the researcher is looking for the information, and many approaches can be used to achieve the research aim and objectives, such as quantitative, qualitative or mixed techniques, and inductive or deductive reasoning (Neville, 2007). The next section gives more details on traditional data collection methods.

6.4 Data collection methods

According to Saunders et al. (2009), Denzin (2006), the use of triangulation in data collection has become popular in recent research. The term triangulation refers to the use of secondary and primary data as the two sources of data collection. Secondary data refers to data published in papers or online, for example, whilst primary data refers to the data collected by the researcher in order to obtain the results needed.

6.4.1 *Secondary and primary data*

Secondary data is the data source used by a study to reach the research aim. This type of data is published and documented by others via many methods, such as journal articles, books, reports, newspapers, etc. (Hox and Boeije, 2005). Furthermore, secondary data helps the researcher to build a theoretical and conceptual framework about certain knowledge as it gives the researcher the tools to find the research gaps. Also, it is considered an inexpensive tool to use to gather data. It is also well known as being time saving compared to primary data. However, one of the disadvantages of secondary data is that it may not be up to date. The relevance of some data will be outdated in certain subjects, and so researchers should be very careful about what data should be used and reviewed. Another of the main disadvantages is that secondary data has not been designed to follow the specific research data collection strategy (Saunders et al., 2009). However, it does offer a number of benefits, as earlier outlined, and so this research will use secondary data to extract risk, success factors and review publications about the relevant subjects.

Primary data is the data collected by the researcher in order to investigate and find a solution to a specific problem. In general, most of the data collected is added to the existing data knowledge where this data, if published, becomes a secondary data resource for other researchers. Questionnaires, surveys and observation are the well-known tools used to collect primary data. Furthermore, the advantages of primary data lie in the fact that it has been designed to reach the research aim and objectives, as well as the data collection strategy is more specific to certain data than the wide data options that are available in secondary data (Hox and Boeije, 2005). One of the drawbacks of primary data is that it is not a cheap tool. Also, collecting data is time consuming as researchers usually spend a lot of time on collecting and designing questions and sometimes they do not get the response they hoped for (Hox and Boeije, 2005, Collis and Hussey, 2013). Questionnaires are used in this study in order to obtain the data needed to answer the research questions.

According to Platt et al. (1992), a questionnaire is "a set of questions on a topic or group of topics designed to be answered by a respondent". They have been used widely in data collection methods used to research the software risk as well as in information technology to gather data (Nakatsu and Iacovou, 2009, Li et al., 2008, Islam et al., 2009, Addison and Vallabh, 2002, Han and Huang, 2007). One of the main advantages of using a questionnaire is that it can gather data from a large number of respondents in a relatively short time (Kombo and Tromp, 2006). Maylor and Blackmon (2005) have noted that a questionnaire has two types of questions. The first are open-ended questions, where respondents are asked to write their opinion about a topic or a group of topics. Their answers can be used as qualitative data. The second are closed-ended questions, where respondents are asked to choose or rate questions. This data can be used as quantitative data in empirical research in social science (Vehovar and Manfreda, 2008).

6.5 Research methods

Qualitative and quantitative are two of the well-known categories of research design. According to Cresswell (1998), ‘What?’ and ‘How?’ are the two main questions that are addressed via a qualitative methodology. Strauss and Corbin (1990) define qualitative research as “any kind of research that produces findings not arrived at by means of statistical procedures or other means of quantification”. Qualitative methods usually use an inductive approach (Gibbs, 2002, Lauri and Kyngas, 2005) which, if there is a of knowledge about a certain topic, is considered a suitable approach to tackle this problem (Elo and Kyngäs, 2008). Furthermore, words and observation are used in qualitative methodology to build up and understand people’s behaviour or certain topics (Amaratunga et al., 2002). Also, Boussabaine (2013) noted that in risk classification and management qualitative techniques are considered to be one of the most suitable techniques. On the other hand, according to Li (2016), “Quantitative is predominantly used as a synonym for any data collection technique (such as a questionnaire) or data analysis procedure (such as graphs or statistics) that generates or uses numerical data”.

A third technique is the combination of both qualitative and quantitative methodologies, which has become popular among researchers. This mixed method started to emerge in the 1990s as a scientific method for achieving the research aim and answering the research questions (Tashakkori and Teddlie, 1998). Mixed methods is defined by Creswell et al. (2003) as using qualitative and quantitative methodologies in research to collect data or in the analysis of data, where the use of the two methods can be at the same time or sequentially as long as data are integrated. Using mixed methods in scientific research helps to gain the strengths from both methodologies and avoid their disadvantages (Amaratunga et al., 2002). The use of mixed methods has become important in complexity research, as the use of only qualitative or quantitative methodologies, according to Creswell (2013), “is inadequate to address this complexity. The interdisciplinary nature of research, as well, contributes to the formation of research teams with individuals with diverse methodological interests and approaches”.

Some of the advantages of using mixed methods includes: first, the questionnaire in the quantitative research can be developed by using data from the qualitative methodology (Amaratunga et al., 2002). Second, the bias in using one method can be overcome or cancelled by using mixed methods due to the fact that the other method could have the strength to eliminate that weakness (Mathison, 1988). Third, mixed methods have been proven in many cases to have the ability and strength to answer the research questions where other approaches were not able to (Amaratunga et al., 2002).

6.6 The rationale for the research design

There are different reasons behind choosing any methodology. One of the main reasons is the fact that it can answer the research question. So, this research has chosen to use mixed methods in order to answer this research questions, and there are many reasons behind this choice. For example, the mixed methods approach (use of quantitate and qualitative) has proven over the years that it can provide a clear view of a research question (Creswell, 2009). Furthermore, according to Morrison et al. (2012), mixed methods enhance the advantages and /or reduce the weaknesses of each methodology. It also helps the researcher to reduce intrinsic bias, which usually occurs when one methodology is applied (Carr and Smeltzer, 2002, Denzin, 1970). It is most likely that using mixed methods strengthens the results and makes them more reliable (Jogulu and Pansiri, 2011, Brewer and Hunter, 1989). Another reason for choosing mixed methods is triangulation, which is the use of two questionnaires (impact and expert) as the most appropriate method to give a more complete overview of a phenomenon (Sale et al., 2002). Mixed methods have also been used for investigation and research in the field of software and IT projects (Ryan and O’connor, 2009, Basirati et al., 2015, Marsavina, 2014, Parry et al., 2011). Maruping et al. (2009) stated that mixed methods should be used when researchers are investigating a phenomenon in software development.

6.7 The research method process

When the decision to use mixed methods in a study is taken, it becomes necessary to identify the steps in this methodology that will help to answer the research questions (Hyman and Yang, 2001, Sila and Ebrahimpour, 2002, Morse et al., 2003). This includes the research design as well as how the data is collected, and the sample attributes. Additionally, it illustrates how the data will be analysed to reach the research aim and objectives. The method in this research contains five stages, as shown in Figure 6-2. The first stage illustrates the research problem statement, aim and objectives. In the second stage, the research reviews the relevant published journal articles and books in order to build up knowledge about the software risks and risk management in software development, as well as exploring the software development methodologies. It also defines and provides an overview of complexity theory, identifying the risks as well as the success factors in software development. In the third stage, the data collection process will be explained in detail by describing how the two questionnaires are designed and validated, and sample response statistics will be provided. The fourth stage explains the data analysis process, statistical descriptive analysis, tests hypotheses and fuzzy network centrality mapping, which present the respondents' point of view . Also, the network centrality and ego networks will be provided from the experts' point of view. The final stage discusses all these results and draws a conclusion Based on this outline, each stage will be discussed in more detail in the next section.

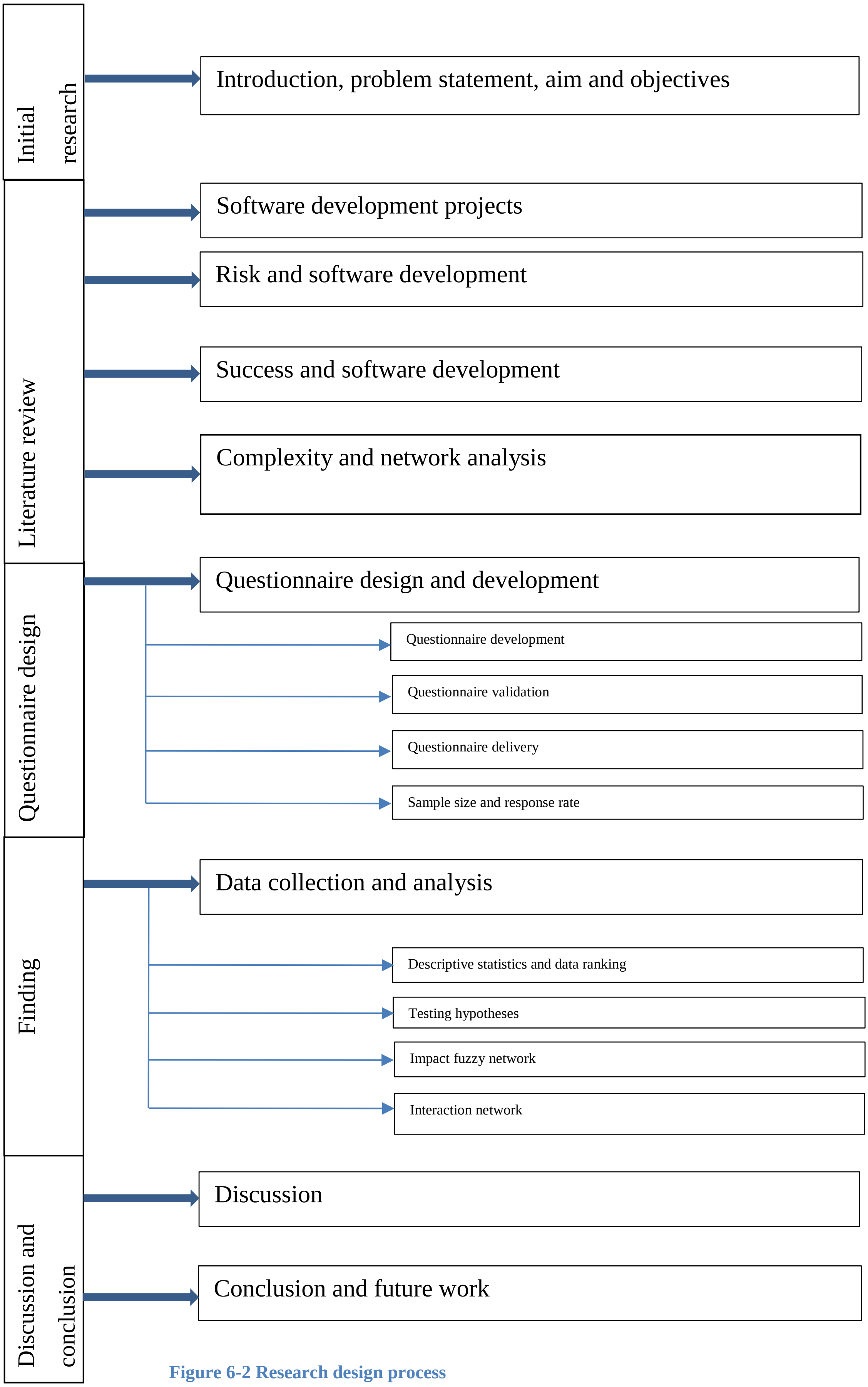


Figure 6-2 Research design process

6.7.1 *Initial research*

In this stage, this research identifies the research questions and problem statement, as well as the research limitations. As result, the aim and objectives that help the researcher to answer the research questions are identified.

6.7.2 *Literature review*

In order to understand the research topic and identify the research questions, this research has collected secondary data through a review of previous literature, using available resources such as journals, books, eBooks, articles and electronic/scientific database libraries provided by the University of Liverpool. Thus, as result of reviewing these studies, the research gap has been determined and accordingly, in order to fill this knowledge gap, the research aim has been set. In addition, this research uses the literature review as a foundation and to explain the new knowledge about the complexity of software projects. The researcher divided the literature review into four sections, as follows:

6.7.2.1 *Software development project*

This is the second chapter in this research. It began by defining software, and then introduced the reader to the concept of software projects by reviewing the highlights of methodologies used in software development projects. In addition, it reviewed software project characteristics.

6.7.2.2 *Risk and software development*

This is the third chapter in the research. It focused on the concept of risk, defining risk and the link between risk and failure of software projects. Additionally, the concepts of risk management have been reviewed. This chapter also reviewed the most important taxonomies of the risks in IT and software projects, especially. the SEI taxonomy, which will be used in this research. In addition, the chapter identified the most important risk factors facing software projects.

6.7.2.3 *Success and software development*

This is the fourth chapter in the research. It focused on the concept of success in projects. Furthermore, it defined success and reviewed the measurement of success in projects. In addition, it identified the success criteria for software projects, detailing four key success criteria – cost, quality, time and scope. Moreover, success factors were identified and classified into thee classes: product engineering, development environment and program constraints.

6.7.2.4 *Complexity and network analysis*

This is the fifth chapter in the research. It aimed to understand the concept of complexity by defining it as well as reviewing complex systems. In addition, characteristics of complex systems were reviewed in detail. Another of the objectives of this chapter was to illustrate the relationship between complexity and the failure of software projects in detail. Also, the chapter reviewed the concept of network analysis through the definition of the term network as well as network topologies. Furthermore, another objective of this chapter was to find the measures that reveal important elements for the network via reviewing the centrality measures in network theory.

6.7.3 *Questionnaire design and development*

Questionnaires or surveys can be defined as the tool or method used to reach identified target respondents in order to obtain certain data in a relatively short time. A survey can be descriptive, exploratory or, in some cases, both (Arlene and Kosecoff, 1985). According to Fowler Jr (2013), questionnaires are one of the most commonly used methods to collect data as they have proved to be suitable for collecting data from many respondents in a short time where each respondent is asked to answerer the same questions. In questionnaires, there is no limit to how many questions can be asked. Instead, it depends on the researcher's understanding of the topic, respondents and what information is needed to obtain certain knowledge (Denscombe, 2014). According to McNeill (1990), the data collected from a questionnaire is usually easy to analyse and it also can be compared to demographic background, other questions or other results, which is one of the advantages of using a questionnaire. Another advantage is that it is one of the easiest and most diverse methods of data presentation like graphs and tables. In addition, Burns (1997) noted that the answers are usually honest because the respondents are anonymised. Also, it gives respondents more time to read the questions carefully and answer them. A questionnaire also gathers more data than other methods. On the other hand, questionnaires have certain disadvantages, such as the difficulty of knowing whether the responded has understood the questions as the researcher intended (McNeill, 1990). Another drawback is that it is almost impossible to interact with a respondent's answers, while researchers who use different methods like interviews are flexible in this matter as they can build other questions upon the respondent's answer (Burns, 1997). Kelley et al. (2003) noted that some of the criteria that a questionnaire should have are:

- It must be clear to read and the researcher must try to eliminate ambiguity in the questions. Also, there are should be clear guidance throughout.
- It must avoid any type of writing that it might make the questions difficult to read.
- Each question should have a number.
- It is advisable to avoid questions with two parts ('double-barrelled' questions).
- Questions should be within the research aim (Denscombe, 2014).
- Questions are easier to analyse via software programs like SPSS.

Above all, the questionnaire method is believed to be the appropriate data collection method to reach this research's aim and answer the research questions, and the extensive literature review was used as the main source of knowledge to write and design the questionnaire. In addition, constructive discussion with the researcher's supervisor has been used in the questionnaire design process, as well as the researcher's previous experience.

The study has used two different questionnaires. The first was sent to people working in the software project field (construct correlation), whilst the second was sent to experts in software projects (dependency matrix questionnaire). Both questionnaires are explained in more detail, as follows:

6.7.3.1 *Construct correlation*

The construct correlation has been divided into two parts (risk factors and success factors) and each part contains three sections. In the first section, the researcher's name and contact information and the aim of the research are provided as well an explanation about the confidentiality of the respondents' personal information. The respondents are asked about their general information – name, phone number, organisation, email address and job title. All these questions optional, except the one that asked if they have worked on a software project, and the one that asked for their job title, as this is essential to the research objectives. Jobs have been divided into two groups (management team and development team).

The second section asks questions about the importance of risk factors on the failure of software project, and respondents were asked to rate a list of variables by using a Likert scale. Likert scales are useful to gather the sensitive differentiation in the respondents' opinion about variables, and have been used in the field of software development by many researches (Tiwana, 2004, Jiang et al., 2004, Levesque et al., 2001, Stewart and Gosain, 2006a, Murphy et al., 2013, Hu et al., 2013, Elzamly et al., 2016). They are also one of the most widely used methods to rate variables in surveys (Li, 2013). In this questionnaire, the first question is about the importance of the risk factors on software projects. The respondents have been given five Likert scale options, as shown below:

Table 6-1 Construct correlation questionnaire

Not important	Slightly important	Moderately important	Very important	Extremely important

A linear Likert scale has been used as the factors listed in the questionnaire have been extracted from the literature. Moreover, all of the factors have been proven to have an impact on the project, so the participants have been asked to rate the impact of each factor using the Likert scale taking into consideration their experience of facing those factors. In the third section, the respondents were asked to rate the influence of a list of risk factors on four criteria (cost, quality, time and scope), based on a four-scale Likert scale (not applicable (NA), low, moderate , high). The same sections and questions have been applied to the success factors in the fourth section

6.7.3.2 *Dependency matrix questionnaire*

The questionnaire for the experts is divided into three section. Section one provides the researcher's name and contact details, the aim of the research and a statement of confidentiality. The second section asks about the experts' general information (name, organisation, contact details and education). The third section uses a matrix design: the rows contain the success factors list and success criteria, while the columns contain the list of risk factors and success criteria. The experts were asked to tick any of the risk factors that have a direct impact on the success factors and success criteria, as shown in Table 6-2.

Table 6-2 Dependency matrix questionnaire

	Efficient project management	Realistic budget	Good change management	Team training	Proper planning
Unclear customer requirements					
Unrealistic schedule					
Problems in testing tools					
Inappropriate design					
Data privacy issues					

6.7.3.3 *Questionnaire validation*

After the questionnaire design had been completed, it was presented to professionals and academics in the field of software and risk management in order to obtain their views in terms of the clarity of the questions, identify and remove any ambiguity in the questionnaire, and gain their overview of the questionnaire design. Based on their notes and feedback, the final version of the questionnaire was updated ready for distribution to the target sample.

6.7.3.4 *Pilot study*

A pilot study has been conducted in this study in order to increase the validity and clarity of the both questionnaires. Aspects like complexity and length of the questions, reliability, and the guidance on how to complete the questions are some of the main points addressed by the pilot study. The importance of a pilot study, according to Creswell (2009), is that it enables the researcher to avoid any faults or defects in the data collection tool. Also, Sheperis et al. (2016) noted that reliability and data validity are two of the main results that researchers gain from deploying pilot studies. Furthermore, some researchers, such as Bradburn et al. (1992), have stated that it is critical to have a pilot study to avoid ambiguous questions. The feedback and notes are used to build an understanding of the time required to complete the questionnaire and participants' inclination to participate in it. As a result, the final version was deployed to respondents. Saunders et al. (2009) noted that the pilot study should help researchers to build questionnaires that are easy for the respondent to understand and complete.

6.7.3.5 *Questionnaire delivery*

Many methods can be used to deliver a questionnaire to respondents, such as delivering it in person, by mail, telephone or online. According to Nair and Adams (2009), one advantage of using an online survey is because it is easy to distribute to the respondents. Another advantage is that this method is time saving. In addition, another main advantage is the ability it gives the researcher to monitor and store data more easily than other methods. Dillman (2000) noted that an online survey increased the "data collection efficiency". Wright and Schwager (2008) noted that it is preferable for academics to use online survey delivery in conducting research. Also, many studies about software projects have used the same online survey as the data collection method (Prasad et al., 2010, Tao et al., 2014, He et al., 2007, Stewart and Gosain, 2006b, Guo and Seaman, 2008).On the other hand, one of its disadvantages is that it does not usually generate a good response rate (Nair and Adams, 2009). In order to limit the disadvantages, this research used two methods of delivering the construct correlation online and hand-delivery. The hand-delivered questionnaire has a large response rate (Ringim and Yussof, 2014). With regard to the expert questionnaire, only an online survey was used as the participants had been preselected and had agreed to fill in the questionnaire, whilst for the construct correlation respondents working in the IT department in government sectors such as the Ministry of Health or those working or who had worked on software projects in the private sector were given the option to complete either the paper-based questionnaire or the online survey.

6.7.3.6 *Population, sample size and response rate*

Population is the target sample, which could be managers, staff, community members, government employees or the general public who have the relevant characteristics to participate in the data collection method (Groves et al., 2009). The researcher has to identify the sample size that should be targeted in order to collect the relevant data. According to Groves et al. (2009), the sample size is usually a small number representative of the much wider targeted population. The response rate, according to Biemer and Lyberg (2003), is one of the main attributes that measure the questionnaire quality. The American Association for Public Opinion Research (AAPOR (2004) defines response rate as “the number of complete interviews with reporting units divided by the number of eligible reporting units in the sample”. Creswell (2009) has noted that the sample’s experience of a phenomenon like risk over the years plays a major role in their participation in the questionnaire. Therefore, this study distributed 300 questionnaires to the targeted sample of professionals in the field of software projects who all are working in Saudi Arabia, and received 107 completed questionnaires, which is a response rate of 35.67%. Watt et al. (2002) noted in their research that that the average response rate to an online survey is 32.6% while for a paper-based survey it is 33.3%. Furthermore, Walstrom and Wilson (1997) noted that 10.4% is the minimum percentage of reliable respondents required in order to obtain data from a certain group. This is also confirmed by Walstrom and Wilson (1997), who found that in their research that 14.4% would be a reliable sample in order to conduct a research. This research has noted that, although the response rate is considered to be reliable, it could have been higher. In addition, Bennett and Nair (2010) mentioned that surveys with a 30% response rate are also considered reliable, whilst Deutskens et al. (2004) observed that the length of the questionnaire is the main factor that affects the response rate – by about 7%, in their finding.

6.7.4 *Data collection and analysis*

The next logical step after the data has been collected is to process this data. Data analysis is considered to be widely different for different research problems. Furthermore, the researcher has to choose the necessary data analysis method and techniques in order to reach the research aim and answer the research questions (Weber, 1990). Each data analysis method provides its own view of the data. The researcher has to employ skill and the ability to use the available tools to his/her advantage as there are no an exact data analysis steps (Hoskins and Mariano, 2004). The analysis should be detailed so that other researchers can understand how the results have been obtained, and also to give them the opportunity to use the same techniques and apply a similar approach in their research (Evaluation and Division, 1996). This research, as has been mentioned, used mixed methods in order to answer the research questions, and divides the analysis into four chapters (descriptive statistics and data ranking, testing the hypotheses, impact fuzzy network and interaction network), which will be described in more detail in the following sections.

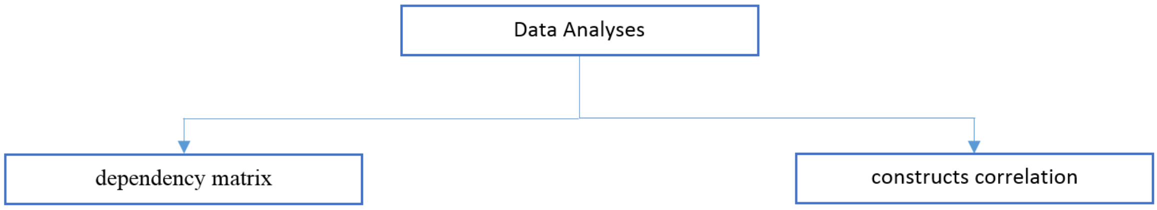


Figure 6-3 Data collection and analysis

6.7.4.1 *Descriptive statistics and data ranking*

In this chapter, the research provides an overview of the participants' demographic information. It is also uses two software packages (IBM Statistical Package for Social Sciences (SPSS) and Microsoft Excel) for the descriptive analyse and data ranking. According to Bryman and Cramer (2011), the benefit of "SPSS is that it will enable you to score and to analyse quantitative data very quickly and in many different ways". The risk factors are ranked based on the average weighted mean, coefficient of variation, standard deviation and severity indices.

6.7.4.2 *Testing the hypotheses*

One of this research's objectives is to investigate whether the two groups of respondents exhibit any significant differences through the research hypotheses. T-test is one of the most well-known and trusted statistical tools used in finding the significant difference between two groups. There are two types: the first is the independent samples t-test where, according McCrum-Gardner (2008), the t-test "is used to compare sample means from two independent groups". The second type is the paired samples t-test, which tests the mean for the same group but if there are differences in the sample circumstance (McCrum-Gardner, 2008). This research uses the independent samples t-test to investigate the significant difference between the management and development teams with regard to the software development project. In order to find the difference between the groups, a significant level of 0.0 to 0.05 has been applied where the hypotheses used in order to justify the statistical differences in the groups' responses are:

H_0 *(p>0.05)*: There is no significant difference among the respondents' ratings for the importance of risk factors on the software project.

H_1 (p < 0.05): There is a significant difference among the respondents' ratings for the importance of risk factors on the software project.

As a result, the factors that have been found to have a significant difference are listed in Chapter7.

6.7.4.3 ***Impact fuzzy network***

Fuzzy congestive mapping (FCM) has been used for the past 20 years; it was introduced by Kosko (1986) and is mainly based on cognitive mapping theory. The fuzzy cognitive map is a graphical representation of elements or factors represented as nodes connected to each other via arrows known as edges (Wildenberg et al., 2010). One of its advantages is that it is easy to understand and it has the ability to describe the interaction between the factors (Papakostas et al., 2015). Fuzzy congestive mapping has been used by many researchers (Wildenberg et al., 2010, Keesang et al., 1996, Stach et al., 2005, Limon et al., 2014, Hossain and Brooks, 2008, Puheim et al., 2015) in order to explore complexity issues or a whole system. According to Puheim et al. (2015), "Its applications cover a wide range of diverse areas such as political and social studies, information technology, robotics, expert systems, engineering, medicine, education, etc.". Furthermore, FCM describes the interactions of all the elements in the complex topic studied, and also shows the importance of specific elements based on their interaction (Wildenberg et al., 2010, Papageorgiou, 2013). Therefore, this research uses FCM to answer the research questions. In order build FCM graphically, this research uses three software tools. The first is SPSS, where the correlations between factors have been used as any factors with a significance of 0.05 or less are connected together. The second is the FCMapper, which has been used by many researchers (Wildenberg et al., 2010, Olazabal and Reckien, 2015). It is used to calculate all indices from the construct correlation and a file is produced that can be used and analysed in the third software tool, Gephi, which. has been used to produce and analyse the network of interaction between the software development elements. Gephi software has been used over the years by experts in analysing complexity networks (Heymann and Le Grand, 2013). Again, SPSS has been used to provide a statistical description of the results from the graph as part of the analysis. Furthermore, the three sections of the analysis will be mentioned in more detail in the as next sections.

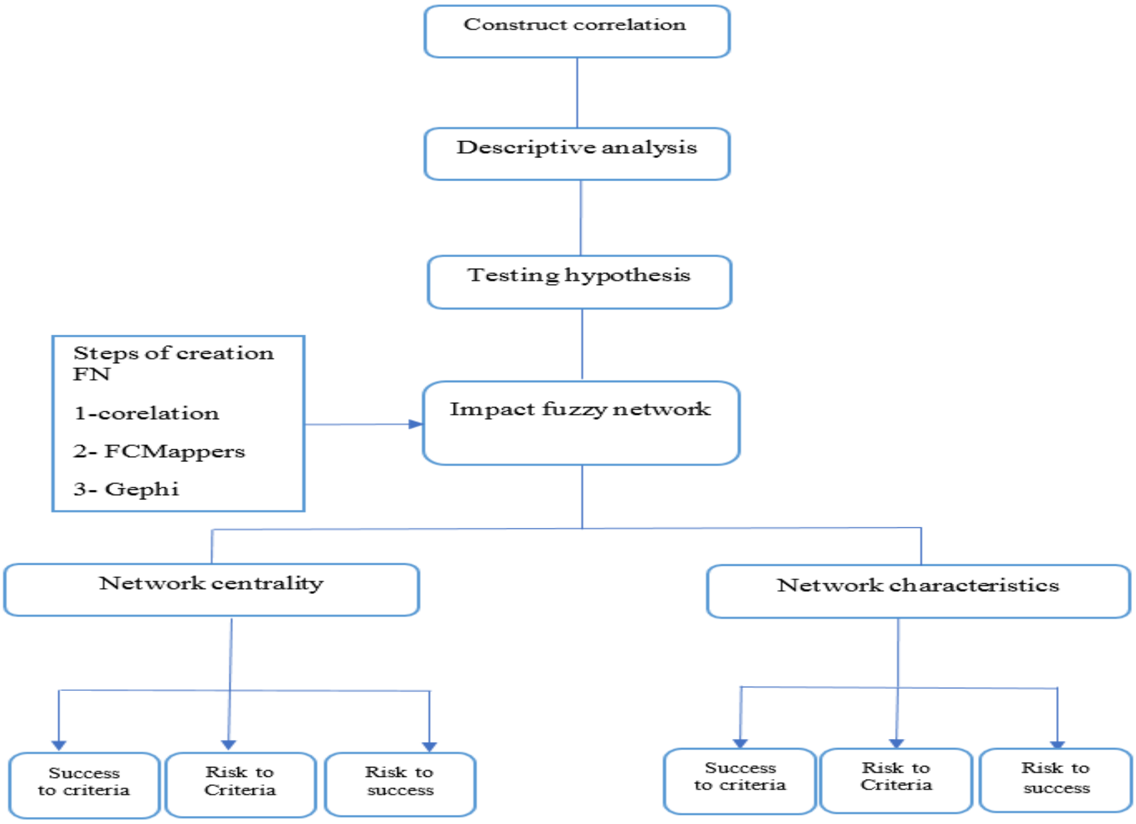


Figure 6-4 Construct correlation analysis diagram

6.7.4.3.1 Network characteristics

In order to investigate the complexity of risk interaction with success factors and criteria in the FCM this research used the results from the Gephi software. The results were then processed via SPSS to come up with network general characteristics like the Minimum, Sum, Mean and Std. Deviation, as general characteristics help to understand the network (Boussabaine, 2013). Moreover, in order to explore in depth the complexity of interaction in this network, the general characteristics have been provided for three sub-networks. First, the general characteristics for centrality measures of risk factors and how they interact with the success criteria have been produced. Then, the general characteristics for the centrality measures of risk factors in relation to success factors and characteristics for the centrality measures and the success factors in relation to success criteria are produced as the second and third sub-networks. More details can be found in Chapter 8.

6.7.4.3.2 Network centrality

The most central factors have also been explored. These factors have been investigated through four measurements. First, in order to identify which factor has the most direct interaction between other factors (‘Degree’ centrality). Second, the most central factors in terms of efficiency and ability to reach other factors (‘Closeness’ centrality). Third, the factors with the most controlling or the shortest paths in the sub-network (‘Betweenness’ centrality). Fourth, the most central factors in terms of their connection to other central factors (‘Eigenvector’ centrality). In addition, overall centrality is mentioned .

6.7.4.4 ***Interaction network***

After the exploring the fuzzy cognitive mapping from the construct correlation, this research investigated the interaction network from the experts’ questionnaire. After producing the interaction of the factors in software development, the research then investigated the real interaction of the risk success criteria with risk and success factors. A group of experts who had worked for at least five years in the software project field were asked to identify if risk factors, success factors and success criteria have a direct influence on each other. As a result, a matrix of interaction was created. According to Stankovic et al. (2013), experts’ judgement is one of the best methods by which to explore a phenomenon about software development projects, and it will provide more advantage if it is mixed with other formal methods. More details of the data analysis process will be explained in the next section.

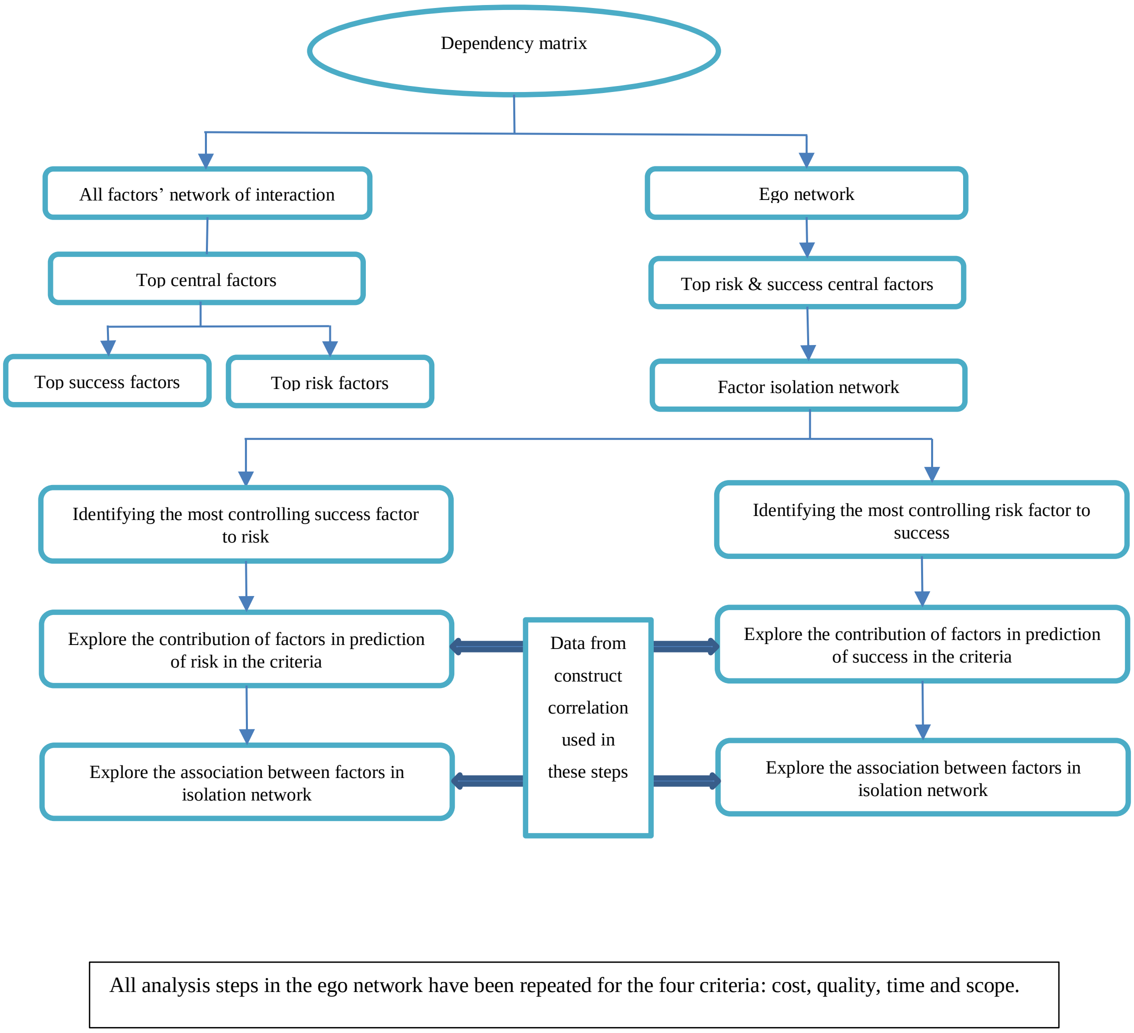


Figure 6-5 Dependency matrix analysis diagram

6.7.4.4.1 Network centrality and factor ranking

In this section, the Gephi software is used to produce a graphical representation of the four success criteria (Cost, Quality, Time and Scope). The centrality measurements for the factors are explored. Several topologies have been applied in order to determine which risk and success factors are the most central in the network as well as how success criteria interact with each other. This type of analysis is called "complete network analysis" (Akhtar, 2014). Details can be found in Chapter 9.

6.7.4.4.2 Ego network criteria

There are only two types of network analysis: complete network analysis, which has been explored in the previous sections, and Ego network analysis (Akhtar, 2014). According to Eleta and Golbeck (2012), "The ego network has become a standard unit of measurement for studying small-scale interactions, or micro-sociology". The ego network can be analysed via looking at its connectivity, characteristics or both (Holes, 1992, Everett and Borgatti, 2005). Ego network topology is applied in this section of the research in order to isolate each criterion and explore several aspects. The first stage is to identify the top centrality risk and success factors in each criterion. The second is to determine the most central success factor influencing the risk factors and how it impacts on the ego network criteria in a later step. A three-step methodology has been applied – see Figure 6-6. First, the two ego network algorithms have been applied to isolate the success factors related to each criterion. Then the degree range algorithm has been applied to determine and give an overview of the graph centrality factors. Then the success factors with high betweenness have been selected as well as the risk factors associated with them in each ego network. The same three-step methodology has been applied to determine the most central risk factor influencing the success factors.

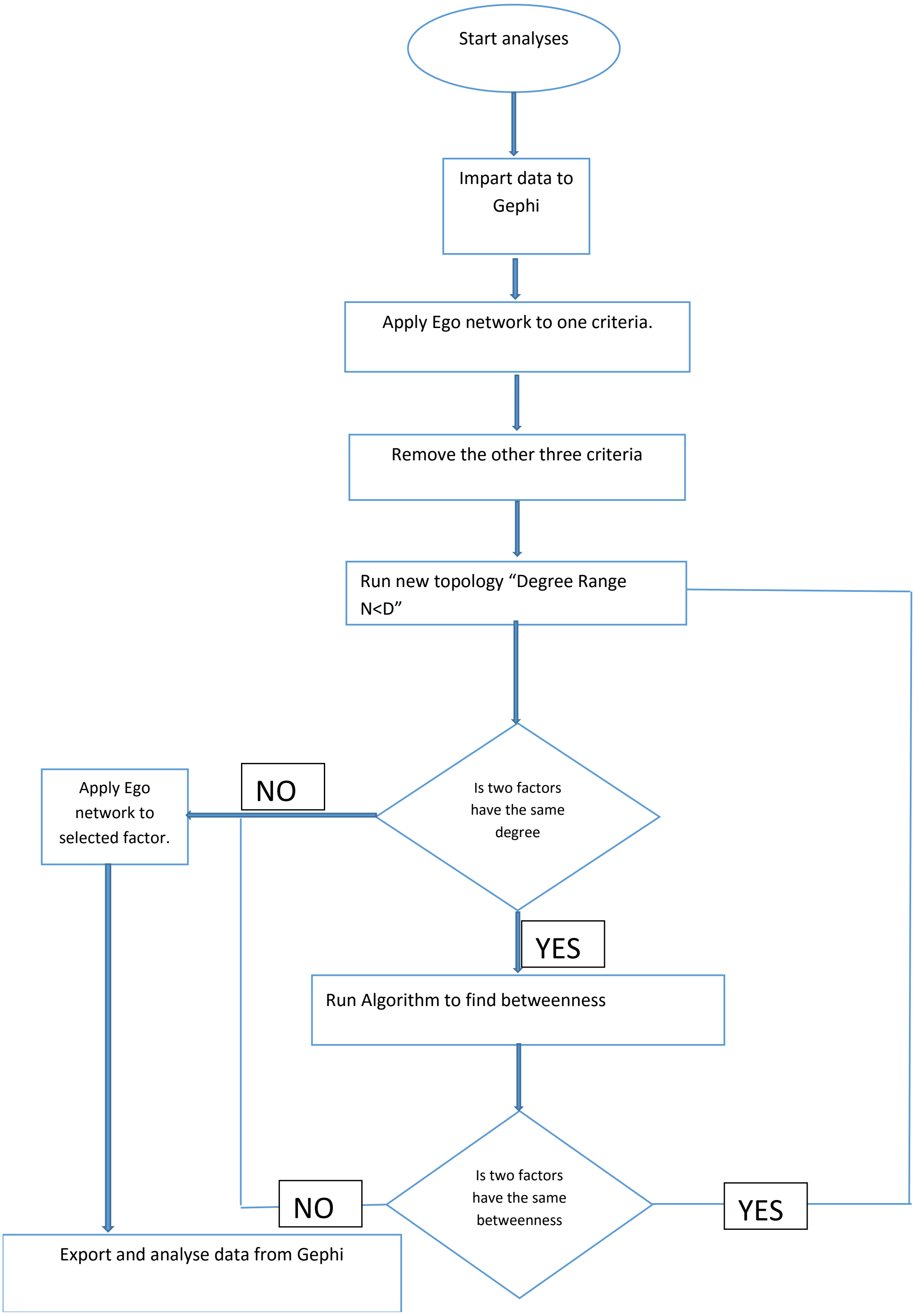


Figure 6-6 Ego network criteria analysis diagram

The third stage models the relationship between the isolated network nodes. The factors that have been found by the experts to be directly connected to the ego network have been selected. After that, the results from the construct correlation are used and stepwise multiple regressions are used to understand and explore how each risk factor will contribute in the prediction of the criteria selected. The same method has been applied to success factors' prediction for the selected criteria.

In the fourth stage, the correlations between the interacted factors in the ego network are used in order to find the significance of the relationship as well as to determine the strongest and the weakest association between the factors. This will be explained in more detail in chapters 10, 11, 12 and 13.

6.7.5 *Discussion and conclusion*

In the discussion chapter, this research discusses the results from the analysis chapter as well as comparing the results to the literature reviewed and related theories in order to answer the research questions. Basically, three areas are discussed in relation to this matter. Firstly, the research hypotheses and factor ranking and some of the descriptive analysis. Secondly, the results from the impact network and how the identified factors impact on software projects. Thirdly, this research discusses the results from the interaction network and impact network and compares them with what previous researchers have concluded.

This research concludes with the results from this exploration and the research novelty. It also identifies the limitations and challenges in this research. Recommendations and the areas that this research believes should be investigated in future are also highlighted.

6.8 Summary

In this methodology chapter, the researcher has provided an overview of the main methods used in scientific research, and highlighted the main reasons behind using the methodology adopted in this research. The research approach and how the data has been analysed are also underlined, as well as how the discussion and conclusion have been approached.

Chapter 7: Descriptive Statistics and Data Ranking

Chapter 7: Descriptive Statistics and Data Ranking

Section one: Descriptive Statistics and classes ranking

7.1 Introduction

This questionnaire has been developed and distributed among practitioners involved in software development projects. This chapter only explains the descriptive statistics of the findings. Further analysis and discussion will be provided in the next chapters.

The first section of the questionnaire (Section A) comprises four questions about the respondents' demographics information. One of the main ones asks about participants' experiences in terms of software development projects.

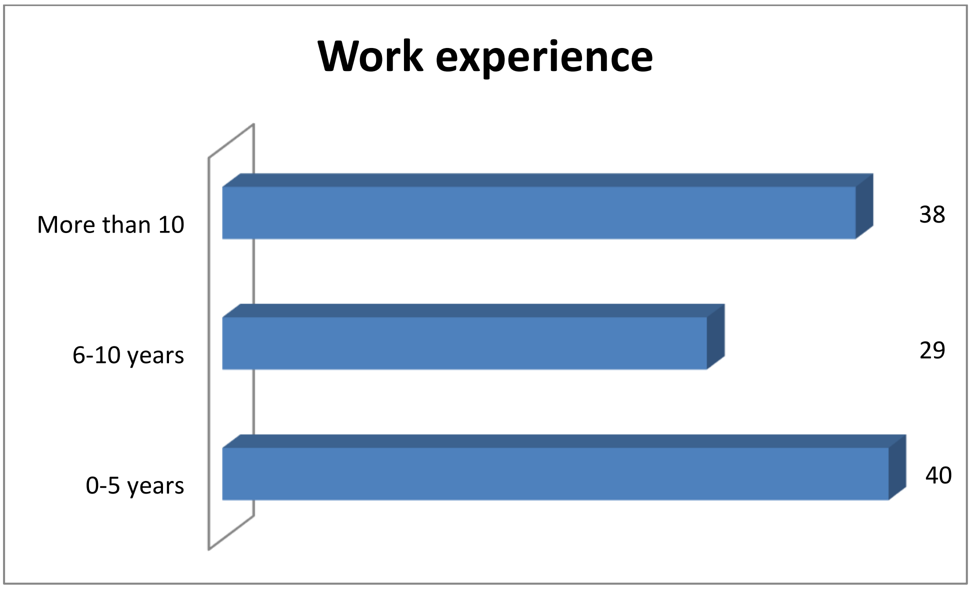


Figure 7-1 Work experience

The findings showed that the majority of professionals in this study have more than 10 years' experience: 40 respondents or 37% of the overall respondents. The number of respondents between six and 10 years' experience was slightly less than the previous category: 38

respondents or 36% of the all respondents, and 29 respondents or 27% of the overall respondents had between 0 and five years' experience.

Another question in Section A was about the respondents' roles in relation to software development projects, where the two main targets of this research are managers and developers, as is shown in Figure 7-2.

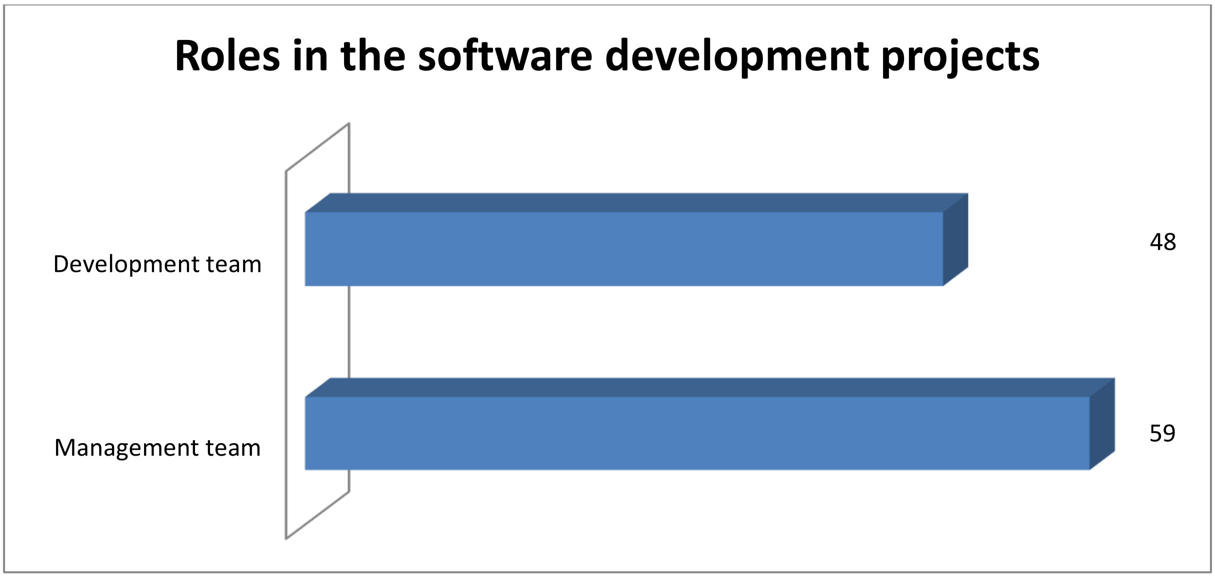


Figure 7-2 Participants' roles in software development projects

The findings showed that the majority of respondents can be grouped into the management team, 59 respondents or 55% of the overall respondents , while 48 respondents (45%) can be placed in the development team.

It is worth mentioning that all the participants are working in a single geographical area, which is Saudi Arabia, as has been mentioned in Chapter 6. Although the sample have enough experience in working on software projects, this research recognised that it is difficult to identify if participants from another geographical area would have significantly different results than the ones obtained by this research. However, it could be an interesting point to be investigated in future research or in order to generalise the results of this research.

7.2 Data ranking

The main reason behind ranking is usually when there is a huge set of date where it becomes necessary to apply ranking in a scientific manner to find and select similar indicators or common themes and trends for the research.

This chapter examines the statistical techniques used to rank the data obtained from the questionnaire survey, which consists of 64 risk factors within three classes. In this study, the SPSS and Microsoft Excel were used for the ranking analysis. The method of evaluation and ranking is based on statistical analysis, as follows (Field, 2005; Morgan et al., 2004; Punch, 2006):

- The average weighted mean
- Standard deviation
- Coefficient of variation
    1. The ratio of standard deviation as a percentage (%) of the mean.
    2. For comparing the relative variability of various responses.
    3. The lower the variation coefficient, the better the variability.
- Severity index
    1. Ranking of the indicators according to their significance.
    2. The higher the percentage (%), the more significant the factor.

All of the above have been used in the ranking. The ranking is based on the questionnaire designed as a result of the literature review and previous research works in this area. It contains 64 risk factors divided into three classes based on SEI taxonomy: Product Engineering, Development Environment and Program Constraints.

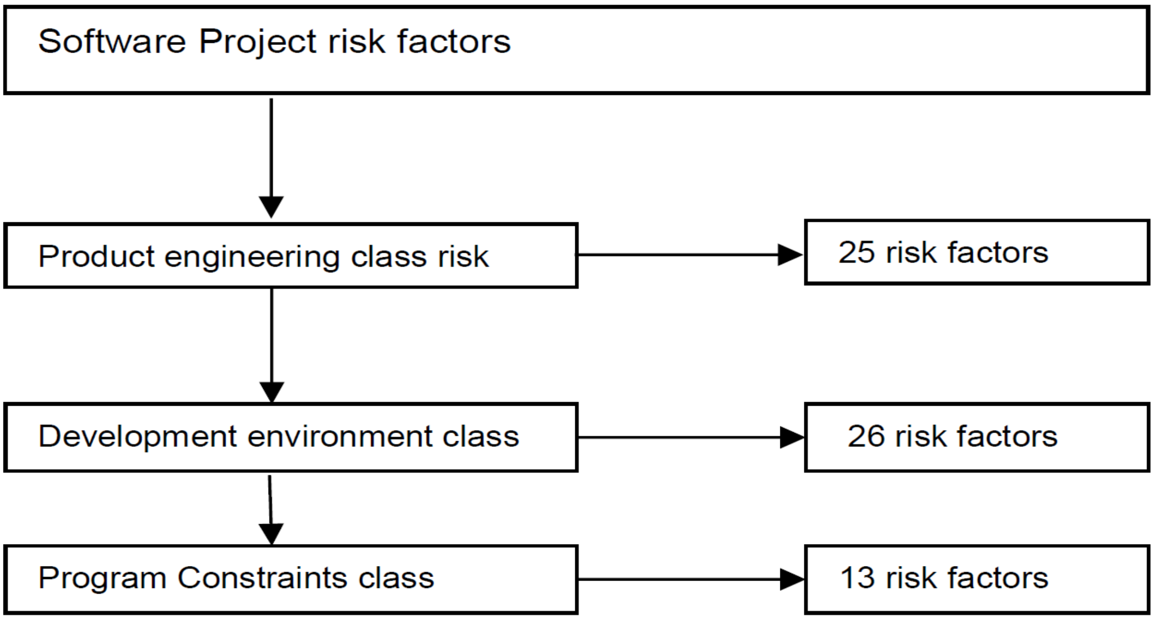


Figure 7-3 Classification of risk factors

7.3 Analysis and ranking of the risk factors

A mean weighted rating for each risk factor is computed to indicate the importance of each indicator, using the equation below:

$$\text{Mean weighted rating} = [\sum (R*F)] / n \qquad \text{equation (1.1)}$$

Where:

R = rating of each risk factor (1,2,3,4,5)

F = frequency of responses

n = total number of responses (n = 107)

Severity index (S.I) measure to rank the indicators according to their significance

Equation (1.2) presents how S.I is calculated.

$$S.I. = \{[\sum(W*F)] / n\} * 100\ \% \qquad \text{equation (1.2)}$$

Where:

W = weight of each rating (1/5, 2/5,3/5,4/5,5/5)

F = frequency of responses

n = total number of responses (n = 107)

The ratio of standard deviation (SO) as a percentage of the mean is called coefficient of variation (COV) and is used to compare the relative variability of the responses.

$$COV = (S / M) * 100\% \qquad \text{equation (1.3)}$$

Where:

S = standard deviation

M = weighted mean sample

The respondents were provided with a list of risk factors formed from the literature review and were asked to rate each risk in terms of its importance on software development projects using the Likert scale of 1-5 (1 – Not important; 2 – Slightly important; 3 – Moderately important; 4 – Important; 5 - Extremely important). Also, the respondents were asked to rate the impact of risk factors on the success criteria (cost, quality, time and scope) using the scale 1-4: (1 – No Impact; 2 – Low ; 3 – Moderate ; 4 – High), where this rating will be used in the interaction network in the next chapter.

7.4 Rating and ranking of risk factors' importance on software development projects

Appendix B shows the ranking results for all 64 risk factors. In the overall ranking, the weighted means are from 2.99 to 4.56 with an overall mean of 3.98. The severity index range is between 59.81% and 91.21%. The top 20 ranked risk factors are dominated by the indicators from the Development Environment Class as it can be seen in Table 7-2. The highest ranked factor was R1ENG - Unclear customer requirements with a mean of 4.56 and severity index of 91.2%. The respondents from both the management side and the development side also rated R1ENG as the highest ranked factor. An overall examination of the first 20 ranked risk factors in Table 7-6 indicates that all have a minimum mean value of 4.10 (which is higher than the overall mean of 3.98) and severity index of 82.05%. This means that the respondents view the first 20 ranked risk factors as important,_believing that these factors have the greatest contribution to software project failure and must be taken into consideration before, during and after completion of a software project.

7.4.1 *Product Engineering class risk factors*

The Product Engineering class consists of 25 risk factors, six of which in this stage were ranked in the first 20 highest indicators, namely R1ENG Unclear customer requirements, R2ENG Unable to meet user requirements, R3ENG Lack of technical skills, R20ENG "Understanding problems of customers , R29ENG Inappropriate technology and R62ENG Unclear or misunderstood scope/objectives. The means for these factors range from 4.16 to 4.56. Also, their severity indices vary from 83.18% to 91.21%. The score of the average weighted mean for all of these indicators is 4.02 and the average severity index is 80.44%, which is very high in comparison with the other classes. The highest ranked factor was R1ENG Unclear customer requirements with a mean of 4.56 and severity index of 91.2%, and it is considered to be the highest ranked indicator for this stage as well as in the overall ranking. R20ENG has an overall ranking of 2nd (out of 64), which is the same ranking results from the development team whilst the management team have ranked it as 7th. Factor R20ENG has a mean of 4.36 and severity index of 87.1% whereas factor R2ENG has an overall ranking of 9^{th,} a mean of 4.24 and severity index of 84.86%. The management team have ranked it as 13th and development team have ranked it as 5th. R3ENG with a mean of 4.21 and severity index of 84.30% has an overall ranking of 11$^{th.}$. The management team have ranked this factor as 9th whilst the development team did not consider this factor to be in their top 20th as they ranked it 21st. R62ENG with a mean of 4.17 and severity index of 83.36% has an overall ranking of 15th. The management team did not consider this factor as in their top 20 as they ranked as 22nd whilst the development team ranked it in 10th place in their top 10. R29ENG has an overall ranking as 17th (out of 64) where the development team and the management team have ranked it out of the top 10 in 13th and 21st places respectively. This factor has a mean of 4.16 and severity index of 83.17%.

Table 7-1 Ranking of risk factors importance on the Product Engineering class

Ref	Mean	Standard deviation	Management team	Development team	Coefficient of variation	Severity index	Overall Ranking
R1ENG	4.56	.675	1	1	14.80922551	91.21495327	1
R2ENG	4.24	.920	13	5	21.6784498	84.85981308	9
R3ENG	4.21	.932	9	21	22.10958188	84.29906542	11
R4ENG	3.74	.904	56	50	24.18869902	74.76635514	55
R5ENG	3.99	.986	31	35	24.70028012	79.81308411	35
R6ENG	4.07	.780	44	7	19.19388932	81.30841121	25
R20ENG	4.36	.780	7	2	17.91185302	87.10280374	2
R21ENG	3.99	.818	41	24	20.50706742	79.81308411	34
R28ENG	4.06	.909	15	41	22.42077632	81.12149533	27
R29ENG	4.16	.779	21	13	18.72721361	83.17757009	17
R31ENG	3.99	1.112	30	39	27.85628041	79.81308411	33
R33ENG	3.96	1.063	49	20	26.83396533	79.25233645	37
R34ENG	4.02	.961	35	26	23.92172967	80.37383178	32
R36ENG	3.97	1.032	26	49	25.98480379	79.43925234	36

R44ENG	3.85	1.035	54	34	26.88667428	77.00934579	47
R45ENG	4.09	.986	16	31	24.08869232	81.86915888	21
R46ENG	3.90	1.027	47	38	26.36027741	77.94392523	43
R48ENG	3.67	1.106	62	42	30.09950497	73.45794393	58
R49ENG	4.08	1.047	28	22	25.64087228	81.68224299	22
R52ENG	4.07	1.049	23	32	25.79187563	81.30841121	26
R53ENG	3.83	1.068	48	55	27.87978059	76.63551402	51
R57ENG	3.93	1.021	33	46	25.95449112	78.69158879	39
R58ENG	3.75	1.182	40	61	31.54942164	74.95327103	54
R61ENG	3.89	1.067	39	48	27.44155685	77.75700935	44
R62ENG	4.17	.916	22	10	21.97974106	83.36448598	15

7.4.2 *Development Environment class*

The Development Environment class consists of 26 risk factors. This class has almost half of the top 20 factors as nine of its risk factors were ranked in the first 20 highest indicators, namely: R22DEV, R32DE, R14DEV, R23DEV, R35DEV, R47DEV, R9DEV, R10DEV and R12DEV. The means for these factors range from 4.15 to 4.32, and their severity index vary from 82.99% to 86.36%. The score of average weighted mean for all of these indicators is 3.96 and the average severity index is 79.22%. Factor R22DEV with a mean of 4.32 and severity index of 86.36% is considered the highest ranked indicator (3rd) for this class. Three more factors, R32DE, R14DEV and R23DEV, were ranked in the top 10, with overall rankings of 5th, 7th and 8th (out of 64) respectively. Factor R32DE has a mean of 4.31 and severity index of

86.17%, whereas factor R14DEV has a mean of 4.30 and severity index of 85.98%, and factor R23DEV has a mean of 4.29 and severity index of 85.79%. Both management team and development team have ranked all those factors in the top 10, as shown in Table 7-2 , except factor R23DEV (Project manager lacks experience), as management has ranked this factor 2nd out of the 64 factors; conversely, the development team has ranked it 16th. The other five factors in the top 20 are R35DEV, R47DEV, R9DEV, R10DEV, and R12DEV, where Factor R35DEV has a mean of 4.21 and severity index of 84.30%, and an overall ranking of 12th. Factor R47DEV has a mean of 4.21 and severity index of 84.11%, and an overall ranking of 13th. R9DEV has an overall ranking of 14th, a mean of 4.17 and severity index of 83.36%. R10DEV has an overall ranking of 16th, a mean of 4.17 and severity index of 83.36%. Factor R12DEV has a mean of 4.15 and severity index of 82.99%, and an overall ranking of 18th.

Table 7-2 Ranking of risk factors importance on the Development Environment class

Ref	**Mean**	**Standard deviation**	**Management team**	**Development team**	**Coefficient of variation**	**Severity index**	**Overall Ranking**
R7DEV	3.95	.935	32	47	23.66386746	79.06542056	38
R8DEV	3.67	1.035	58	58	28.17958187	73.45794393	59
R9DEV	4.17	.947	12	23	22.70877424	83.36448598	14
R10DEV	4.17	.874	20	12	20.96830472	83.36448598	16
R11DEV	4.08	.933	25	25	22.84105163	81.68224299	23
R12DEV	4.15	.930	24	17	22.40336916	82.99065421	18
R13DEV	4.05	1.085	27	30	26.80968385	80.93457944	30

R14DEV	4.30	.838	8	9	19.49156678	85.98130841	7
R22DEV	4.32	.831	3	6	19.24568264	86.35514019	3
R23DEV	4.29	.911	2	16	21.24249935	85.79439252	8
R25DEV	2.99	1.349	64	64	45.11791336	59.81308411	64
R26DEV	4.06	.940	38	14	23.17535663	81.12149533	29
R27DEV	4.06	.920	29	27	22.67509336	81.12149533	28
R32DEV	4.31	.915	6	4	21.2466882	86.1682243	5
R35DEV	4.21	.890	11	18	21.1270498	84.29906542	12
R37DEV	3.45	1.159	63	62	33.61724042	68.97196262	63
R38DEV	3.91	.976	34	52	24.99515631	78.13084112	42
R40DEV	3.92	.982	42	40	25.08035008	78.31775701	40
R42DEV	3.89	1.119	36	56	28.77385336	77.75700935	45
R47DEV	4.21	.939	10	19	22.32903053	84.11214953	13
R50DEV	3.88	.978	45	44	25.2200389	77.57009346	46
R54DEV	3.83	1.041	46	53	27.17959373	76.63551402	50
R55DEV	3.79	1.026	50	54	27.02693131	75.88785047	53
R56DEV	3.91	1.086	37	43	27.80501843	78.13084112	41
R60DEV	3.84	1.065	52	37	27.73448781	76.82242991	49
R64DEV	3.60	1.288	55	63	35.78328858	71.96261682	61

7.4.3 *Program Constraints class*

The Program Constraints class consists of 13 risk factors. The means for these factors means range from 3.56 to 4.31, and their severity indices vary from 71.21% to 86.17%. The score of average weighted mean for all of these indicators is 3.95 and the average severity index is 79.08%, which is very high in comparison with the other classes. For the Program Constraints class, five of its 13 factors were ranked in the 20 highest ranked indicators; three of these (R41PR, R16PR and R18PR) in the top 10. The factor R41PR has an overall rank of 4th out of 64, a mean of 4.31 and severity index of 86.17%. Management and development teams have both ranked this factor in the top 10, as 4th and 8th respectively. Factor R16PR has an overall rank of 6th with a mean of 4.30 and severity index of 85.98%. The management team ranked this factor as 5th in the top 10 and the development team ranked it as 11th. Factor R18PR has an overall rank of 10th with a mean of 4.21 and severity index of 84.30%. The development team ranked this factor as 3rd in the top 10, whilst the management team ranked it as 19th. The last two factors in the top 20 are R17PR, and R30PR. Factor R17PR has a mean of 4.10 and severity index of 82.06%, and an overall ranking of 19th, whilst Factor R30PR has a mean of 4.10 and severity index of 82.06%, and an overall ranking of 20th out of 64.

Table 7-3 Ranking of risk factors importance on the Program Constraints class

Ref	Mean	Standard deviation	Management team	Development team	Coefficient of variation	Severity index	Overall Ranking
R15PR	4.04	.921	43	15	22.80380236	80.74766355	31
R16PR	4.30	.871	5	11	20.26197594	85.98130841	6
R17PR	4.10	.910	14	29	22.19139273	82.05607477	19
R18PR	4.21	1.000	19	3	23.73132997	84.29906542	10
R19PR	3.85	1.097	51	36	28.49519868	77.00934579	48
R24PR	3.56	1.199	61	60	33.66932618	71.21495327	62
R30PR	4.10	1.063	17	28	25.91910815	82.05607477	20
R39PR	3.69	1.085	60	51	29.39544681	73.8317757	57
R41PR	4.31	.905	4	8	21.00612221	86.1682243	4
R43PR	3.64	1.283	59	59	35.20219696	72.89719626	60
R51PR	4.07	.997	18	33	24.47194636	81.4953271	24
R59PR	3.70	1.048	57	57	28.31834925	74.01869159	56
R63PR	3.81	1.074	53	45	28.16539625	76.26168224	52

Section two: Software practitioners' perceptions of the importance of risks and their impact on software development projects

7.5 Introduction

Most of the previous studies undertaken explore the relationship of risk factors to the overall project performance and correlate these issues within the perspectives of the project team. This section explores the respondents' years of experience as well as explores the influence of the respondents' years of experience in rating risks. Then the software practitioners' perceptions of risks and their importance on software development projects are tested using a t-test.

7.6 Findings

7.6.1 *Experience*

Table 7-4 Respondents' experience

Respondents	Years of experience in software development projects			
	0-5	6-10	More than 10	Number of respondents
Management team	16	15	28	59
Development team	24	14	10	48
Total	40	29	38	107
%	37.38%	27.1%	35.51%	

It can be seen that more than half (62.61%) of the respondents had more than six years of experience in software development practice. Furthermore, 35.51% have more than 10 years' experience; the majority of these came from the management team with 28 respondents, whilst 10 respondents were from the development team. Twenty-nine respondents (27.1%) had between 6 and 10 years of experience; they were almost equally divided between the two groups (management team with 15 respondents and development team with 14 respondents. Forty (38%) respondents had between 0 and five years of experience, and this category is slightly larger than the others. It can therefore be said that the respondents have good working knowledge and insights related to software development projects and processes in terms of years of experience. The wealth of experiences among the respondents was very relevant and significant in justifying the responses that were given in the questionnaires. This may give reasonable support for the arguments in this study.

7.6.2 *Overall observation*

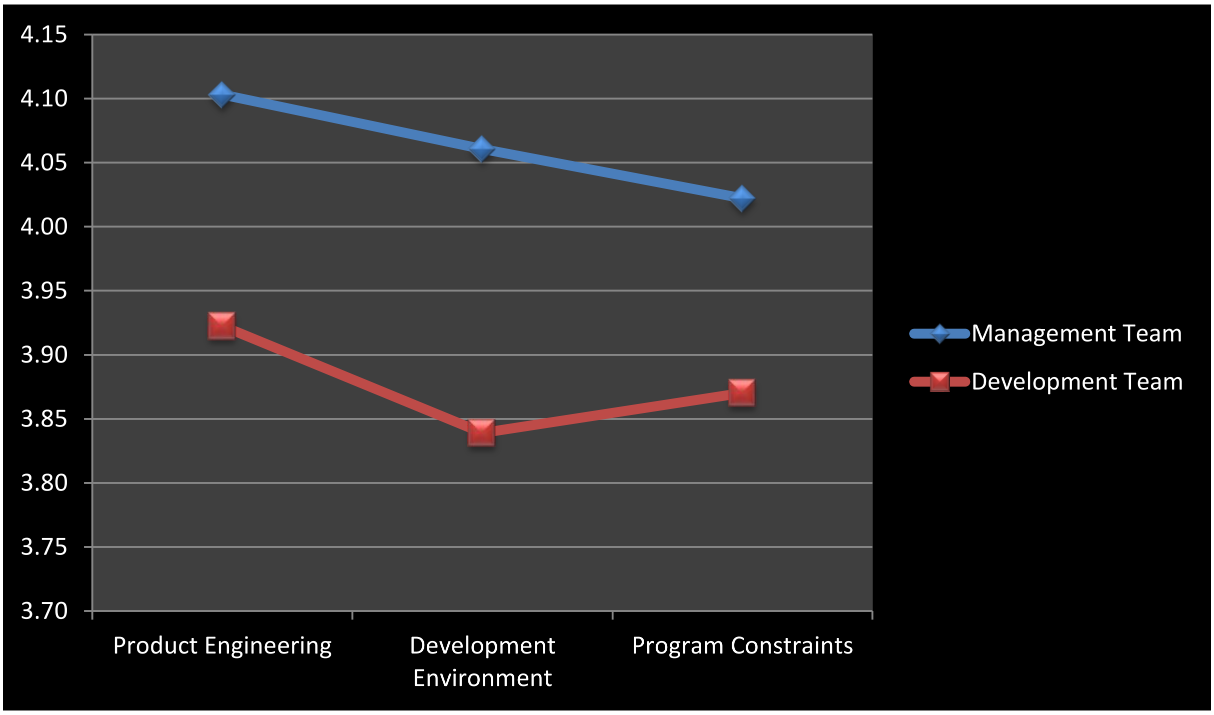


Figure 7-4 The average rating for likelihood of occurrence of risk factors

Table 7-5 Respondents' average mean

Software development class	Average mean	
	Management Team	Development Team
Product Engineering	4.10	3.92
Development Environment	4.06	3.84
Program Constraints	4.02	3.87

From the graph above it can be seen that the Product Engineering class has been identified by participants as the class with the risk factors that have the most impact on software projects, where, in terms of its impact, the management team has rated this class with an average mean of 4.10 while the development team has rated it with an average of 3.92. It is interesting to find that Development has been rated by the development team as the lowest class in terms of its impact importance, although it has been recognised as containing factors from the top 20 ranked risk factors. Although the difference is not significantly great, a reason for it could be due to its higher number of factors compared to the Program Constraints. On the other hand, the management team has rated it the second most impactful class in software projects. Managers have rated Program Constraints as the least important class in software projects in terms of its risk impact, while the development team has rated it with an average of 3.87, which makes it the second most impactful class. It is also noticeable that the management team always gives a higher rating than the development team, rating all the classes with an average mean between 4.02 and 4.10, while the development team has rated all the classes with an average mean between 3.87 and 3.92. The main reason for that could be because the management team is usually involved in all or most of the classes and stages of a software project, and so could have a greater awareness of the consequences to such projects.

7.7 Top 20 rankings

Table 7-6 Top 20 rankings

Stages	Risk factors	Ref	Overall		Management Team		Development team	
			Mean	SD	Mean	SD	Mean	SD
Product Engineering	Unclear customer requirements	R1ENG	4.56	.675	4.61	.588	4.50	.772
	Unable to meet user requirements	R2ENG	4.24	.920	4.31	.915	4.17	.930
	Lack of technical skills	R3ENG	4.21	.932	4.37	.786	4.02	1.062
	Requirement creep (constant changes in the requirements)	R6ENG	4.07	.780	3.98	.799	4.17	.753
	Understanding problems of customers	R20ENG	4.36	.780	4.42	.700	4.27	.869
	Inappropriate design	R28ENG	4.06	.909	4.25	.709	3.81	1.065
	Inappropriate technology	R29ENG	4.16	.779	4.20	.714	4.10	.857
	Size of the project	R33ENG	3.96	1.063	3.92	1.087	4.02	1.041
	Excessive error detection	R45ENG	4.09	.986	4.25	.843	3.90	1.115
	Unclear or misunderstood scope/objectives	R62ENG	4.17	.916	4.20	.805	4.13	1.044
Development Environment	Inadequate infrastructure	R9DEV	4.17	.947	4.31	.793	4.00	1.092
	Unrealistic schedule	R10DEV	4.17	.874	4.22	.767	4.10	.994
	Communication gaps	R12DEV	4.15	.930	4.20	.846	4.08	1.028
	Inefficient team capability	R14DEV	4.30	.838	4.42	.700	4.15	.967
	Improper planning	R22DEV	4.32	.831	4.44	.676	4.17	.975
	Project manager lacks experience	R23DEV	4.29	.911	4.46	.773	4.08	1.028
	Lack of motivation	R26DEV	4.06	.940	4.02	.861	4.10	1.036
	Lack of top management commitment and support	R32DEV	4.31	.915	4.42	.855	4.17	.975
	Unavailable customer contact	R35DEV	4.21	.890	4.34	.801	4.06	.976

	Design is skipped or is created after code is written	R47DEV	4.21	.939	4.34	.843	4.04	1.031
Program Constraints	Staff turnover	R15PR	4.04	.921	3.98	.938	4.10	.905
	Unrealistic budget	R16PR	4.30	.871	4.44	.623	4.13	1.084
	Resource insufficiency	R17PR	4.10	.910	4.25	.843	3.92	.964
	User resistance	R18PR	4.21	1.000	4.22	1.018	4.21	.988
	Market demand obsolete	R30PR	4.10	1.063	4.24	1.040	3.94	1.080
	Lack of staff experience	R41PR	4.31	.905	4.44	.749	4.15	1.052
	Data privacy issues	R51PR	4.07	.997	4.22	.832	3.90	1.153

For the purpose of this chapter and for a more manageable discussion, the risk factors rated by the respondents were ranked and the top 20 ranked risks by each category of practitioners are shown in Table 7-6. The full rankings of the risk factors were already highlighted in section one (Descriptive Statistics and classes ranking). The risk factors in the Development Environment and Product Engineering classes dominate the top end of the ranking. The former has nine out of the top 20, and is the class with the most top factors. Although Product Engineering has slightly more top factors with six compared to Program Constraints with five, the top two ranked factors are from Product Engineering and this class has four out of the top 10 top ranked factors, while Program Constraints has two factors in the top 10. Furthermore, it can be said that these 20 rated risk factors are non-technical risks as they can be categorised as organisational or project management related. Although the results from the perspectives of the management team and the development team tend to agree with the results of some of the previous research and literature reported, there are still a few different views.

7.8 Test of difference

There are two types of t-tests: the independent-samples t-test and the paired-samples t-test. The independent samples t-test is used to examine the differences between two types of group (Frude, 1987) and the paired-samples t-test is used to compare the mean results in different conditions but in the same group (Pallant, 2005). The independent-samples t-test was applied in this study to compare the mean values of two groups (management team and development team). A T-Test analysis was conducted in order to justify the statistical differences in the groups' responses. Using the SPSS software and with a significance level of 0.05 (Pallant, 2005), the hypothesis test is:

H0 ($p > 0.05$): At least one of the group's ratings for the risk factors' importance is significantly different from at least one of the other groups.

H1 ($p < 0.05$): There are no significant differences among the respondents' ratings for the importance of the risk factors.

An independent t-test was conducted to determine if a difference existed between the mean risk factors of the management and development teams. From the table in Appendix B the value in the Sig. (2-tailed) column represents the significant difference in the two categories chosen (management team and development team). Since the respondents were asked to give a rating between the values of 1 and 5, there seems to be a not very significant difference between the groups' responses. However, the table shows that there is a statistically significant difference between the management team and the development team in the mean scores for factors R28ENG Inappropriate design and R23DEV Project manager lacks experiences ([R28ENG = .016, $p <.05$] & [R23DEV = .034, $p <.05$] . Therefore, H1 was rejected. Also, these results show that the significantly different views on the risk factors are in two classes, Product

Engineering and Development Environment. The t- test for Program Constraints has rejected the H0 in this class.

7.9 Summary

Ranking helps researchers to indicate which risk factors are more important in terms of their impact on the success or failure of a software project. Identifying the most important factors helps decision-makers to build their plans and strategies to avoid or control these events. In this chapter, rankings based on severity index, average weighted mean and standard deviation of each risk factor were used in order to determine the degree of significance for risk factors in the context of software development projects; the results show their impact on each class as well. The results show that the Development class has the largest number of factors that have the most significant contribution to failure of software projects compared to the other classes. Furthermore, as can be seen from the Table 7-6, there were some agreements and disagreements among the practitioners in the rating and ranking of the risk factors. This will be discussed in greater detail in Chapter 14. In section two, the research has explored the practitioners' perceptions on the influence of risks and their impact on software development projects via using a t-test for statistically descriptive analysis, which showed that there are significant differences in two classes of risk.

Chapter 8: Impact network characteristics and top central factors

Chapter 8: Impact network characteristics and top central factors

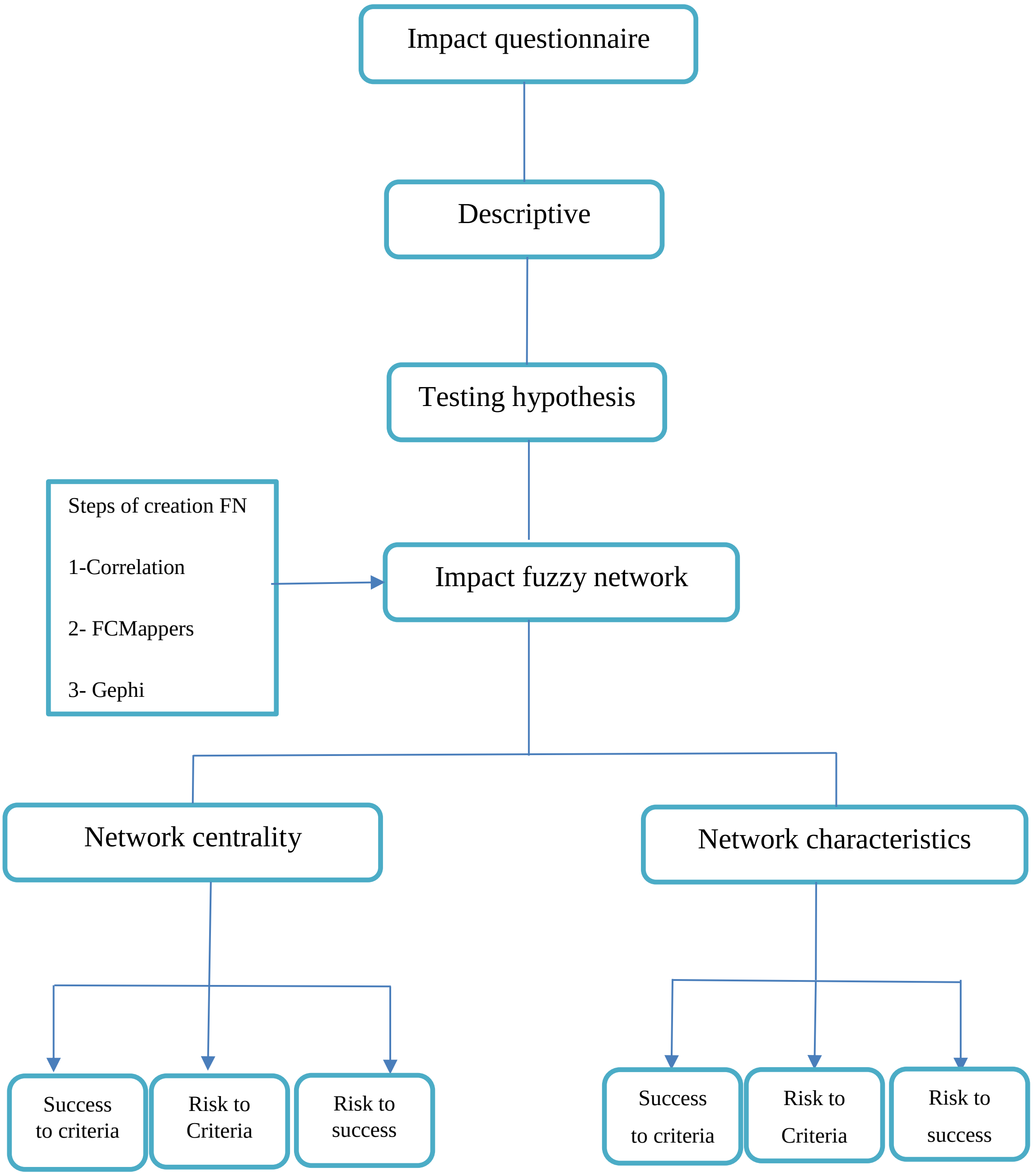


Figure 8-1 Chapter 8 Construct correlation analysis diagram

Section one

8.1 Introduction

In this section, the impact network general characteristics are analysed in detail in terms of three sub-networks. First, the risk factors related to success criteria network characteristics is analysed. Second, the risk and success factors network are analysed. Finally, the success factors and success criteria sub-network characteristics are the third network to be analysed. Section two will focus on the top central factors in each sub-network.

8.2 Risk factors influencing success criteria

Table 0-1 Risk factors related to success criteria network characteristics

Characteristics	Centrality measures			
	Degree	Closeness	Betweenness	Eigenvector
Number of nodes	68	68	68	68
Minimum	5	0	0	0.2393
Maximum	84	1.8358	47.5166	1
Sum	1998	27.1045	409	50.4997
Mean	29.38235	0.398595	6.014706	0.742642
Std. Deviation	23.6738	0.611976	10.79443	0.163874
Variance	560.449	0.375	116.52	0.027

This sub-network focuses on only the interactions that occur between the risk and success factors in order to determine the network's main characteristics. There are 68 nodes in the network, and the density of this network is 0.219. Furthermore, the average degree centrality is 29.38, which indicates that, on average, each risk node is connected to 43.21% of the total number of risk nodes available in the network. Resource insufficiency risk is the most connected node in this network. It is also connected to many central nodes, which can be supported by its eigenvector value of 0.94. The average betweenness in this network is considered to be low, which suggests that this network belongs to a small word network. The maximum distance between two nodes is 1.8358, which is very small. This indicates that the risk generators in software development are well connected and their impact can spread sharply through the network to the other factors.

8.3 Risk factors influencing success factors

Table 0-2 Risk factors related to success factors network characteristics

Characteristics	Centrality measures			
	Degree	Closeness	Betweenness	Eigenvector
Number of nodes	102	102	102	102
Minimum	3	0	0	0.153
Maximum	110	1.861	110.984	1
Sum	1998	22.168	623	66.709
Mean	19.58824	0.21734	6.10784	0.65401
Std. Deviation	24.72717	0.513716	17.39653	0.176632
Variance	611.433	0.264	302.639	0.031

This network is the second sub-network created from the constrictions correlation in order to investigate the network characteristics. It presents the interaction and influence between risk factors and success criteria. This research has found that the degree average in this network is 19.59, which means that the average risk node is connected to 19.08% of the total nodes. The maximum degree connection is 110 and the minimum are subcontracting and Inappropriate technology with a degree connection of 3. The most central node in relation to degree centrality is R14DEV Inefficient team capability; this is also supported by its eigenvector value of 0.9. This node is not only connected to many other general nodes but is also connected to many central risk nodes.

Closeness average in this network is 0.22, which means that this network is well connected where risk factors are close to each other. Furthermore, taking into consideration that the more central a node is the lower its total distance ‘closeness value’ from all other nodes, the most central node factor is Unrealistic resource planning as it has the minimum closeness in this network with 1.05.

Betweenness centrality shows the shortest path between a node and the other nodes in the network; the average shortest path in this network is 6.1. Inefficient team capability has the maximum number of shortest paths in this network, making it the most central node. It has the ability to control the flow of risk impact between two nodes. Therefore, the connection length or distance between two nodes can be considered as the reciprocal value of the connection weight in the case of closeness and betweenness centrality (Bohn et al., 2009; Newman, 2001a, 2001b).

8.4 Success factors influencing success criteria

Table 0-3 Success factors related to success criteria network characteristics

Characteristics	Centrality measures			
	Degree	Closeness	Betweenness	Eigenvector
Number of nodes	42	42	42	42
Minimum	19	0	0	0.7608
Maximum	65	1.0976	1.2118	1
Sum	1998	25.6341	26	39.9702
Mean	47.57143	0.610337	0.619048	0.951672
Std. Deviation	19.88644	0.510037	0.544462	0.043007
Variance	395.47	0.26	0.296	0.002

The third sub-network created by this research aims to investigate the network's main characteristics in the interaction network between only the success factors and success criteria. The degree centrality average in this network is 47.57, which means this network has a very large number of direct connections terms of the degree centrality. The maximum degree connection is 65 and the minimum is 19. The most central nodes in relation to degree are S4DEV, S7DEV, S10ENG, S11DEV, S13DEV, S14DEV, S16DEV, S17DEV, S18DEV, S19PR, S20DEV, S22ENG, S23PR and S24DEV

Closeness average in this network is 0.61, which means that this network is well connected as risk factors are close to each other. Furthermore, taking into consideration that the more central a node is the lower its total distance from all other nodes, the most central nodes factors are S27PR, S31DEV, S32DEV, S33DEV, S36PR and S37DEV as they have the minimum closeness in this network.

Betweenness centrality shows the shortest path between a node and the other nodes in the network; the average shortest path in this network is 6.2. S4DEV, S7DEV, S10ENG, S11DEV, S13DEV, S14DEV, S16DEV, S17DEV, S18DEV, S19PR, S20DEV, S22ENG, S23PR and S24DEV have the maximum number of shortest paths in this network, making them the most central nodes, with the ability to control the flow of risk impact between two nodes.

Section two

8.5 Top central factors

The concept of centrality addresses the issue of how strategic the position of an entity within the network is. Usually, centrality is measured as a property of a single node within the network to evaluate the ‘reachability’ or the ‘importance’ of this node. Degree and closeness centrality suggest a more ‘local’ measure of a node’s centrality within the overall graph, whereas the betweenness and the eigenvector indicate a more global centrality measure.

8.5.1 *Top central factors’ analyses for Risk factors influencing success factors*

Table 8-4 shows that Inefficient team capability is the most central factor in terms of degree as it is directly connected to the largest number of factors. Thus, it is considered to be the most central factor in terms of direct impact on other risk factors or, in other words, its impact can spread directly to the majority of network risk factors and success factors. It should also be noted that this factor is in close proximity to the rest of the risk factors; this is supported by its closeness centrality value, which is 1.05, which reflects the spread time and speed of this factor’s impact on the other factors. In addition, to what has been mentioned inefficient team capability is also the most central factor in terms of its control of other risk and success factors’ interactions with and impacts on each other. This is supported by its value in betweenness of 110.98, which also makes this factor the most central one in this network. Although this factor is not considered one of the top five central factors in terms of connection to many other central factors, it is connected to many highly weighted and important factors; this is supported by its eigenvector value of 0.9. In general, this factor is considered the most central factor for this network.

Unrealistic resource planning and Conflicts among team members are the second and third most central factors in this network as they share the same characteristics as Inefficient team capability but with lower values. It is worth mentioning that unrealistic resource planning has the same distance to other factors as inefficient team capability, which emphasises its efficiency and ability to reach other factors as well as its independence within the network.

Communication gaps is consider the fourth central factor in the risk and success factors network in relation to its direct interaction with other factors as well as its proximity to other factors, which is supported by its degree and closeness centrality values of 85/1.25. However, it is worth mentioning that it does not have much control over the other factors' interactions, as the number of its shortest paths is notably small in comparison to other central factors, which could be the result of its closeness to more central factors that could provide the shortest path to the other factors. Taking into consideration that it is not connected to many highly weighted factors as its Eigenvector value is small with a value of 0.55, this factor is the least central factor of the central factors in relation to eigenvector value.

Inappropriate development process/methodology is considered the fifth central factor in the risk and success factors network in relation to its direct interaction with other factors as well as its average shortest path between factors, as it has the ability to control the interaction between risk and success factors; this is supported by its betweenness centrality value of 45.9. It is worth mentioning that this factor's efficiency in terms of distance to other factors in the network is questionable; this is supported by its higher closeness value of 1.29. Inappropriate development process/methodology is connected to many important risk and success factors in terms of their weight in this network. This is almost the same result as for R8DEV Problems with new technology, although R8DEV is slightly more efficient and able to reach, 'impact', other factor as well as having more independence within the network than Inappropriate development process/ methodology. In addition, it is not connected to many highly weighted factors as its Eigenvector value is small with a value of 0.6. Therefore, Problems with new technology is considered a more central factor than Inappropriate development process/methodology as the former is in the top five central factors in terms of degree, closeness and betweenness.

Factors Resource insufficiency, Government, Inappropriate design, Market demand obsolete and Lack of adequate security technologies are considered to be the most central nodes in terms of their connections to highly weighted factors in the network.

Table 0-4 Top central factors for Risk factors influencing success factors

Factor id	Factor names	Degree centrality	Closeness	Betweenness	Eigenvector
R14DEV	Inefficient team capability	110	1.05	110.98	0.9
R11DEV	Unrealistic resource planning	109	1.05	82.28	0.87
R13DEV	Conflicts among team members	99	1.13	51.8	0.73
R12DEV	Communication gaps	85	1.25	39.76	0.55
R7DEV	Inappropriate development process/ methodology	84	1.29	45.9	0.8
R8DEV	Problems with new technology	84	1.26	41.97	0.6
R17PR	Resource insufficiency	15	0	0	1
R24PR	Government factors	15	0	0	1
R28ENG	Inappropriate design	15	0	0	1
R30PR	Market demand obsolete	15	0	0	1
R49ENG	Lack of adequate security technologies	15	0	0	1

8.5.2 *Top central factors' analyses for Risk factors influencing success criteria*

Table 8-5 shows Resource insufficiency is the most central factor in terms of degree as it is directly connected to the largest number of factors. Thus, this factor is considered a central factor in terms of direct impact on other risk factors or, in other words, its impact can spread directly to the majority of network risk factors and success criteria. It should also be noted that this factor is in close proximity to the rest of the risk factors; this is supported by its closeness centrality value of 1.03, which reflects the spread time and speed of this factor's impact on the other factors. With regard to its betweenness, Resource insufficiency is considered the second most central factor as its betweenness value is 42.04, which also makes this factor the most central factor in this network. Furthermore, its Eigenvector value is 0.9, which shows it is connected to many highly weighted and central factors Although this factor is not considered one of the top five central factors, in general, it can be considered to be the most central factor for this network.

Inefficient team capability is the second most central factor in this network as it is directly connected to the second largest number of factors. Thus, this factor is considered a central factor in terms of direct impact on other risk factors or, in other words, its impact can spread directly to the majority of network risk factors and success criteria. This factor has a closeness value of 1.06, which means that it has the efficiency and ability to reach other factors as the distance to other factors is considerably small. Inefficient team capability has the greatest control of communication and interaction between risk factors and success criteria in this network as it has the largest number of shortest paths available between other factors; this is supported by its betweenness value of 47.52. Although this factor is not considered one of the top five central factors in terms of connection to highly weighed factors, it is connected to many highly weighted and important factors; this is supported by its eigenvector value of 0.88.

Unrealistic resource planning shares exactly the same characteristics as Inefficient team capability but with less control of the other factors' interactions with and impact on each other, as it has a betweenness value of 31.88.

Understanding problems of developers is considered the fourth central factor in the risk and success criteria network in relation to its direct interaction to other factors as well as it close distance to other factors supported by it degree and closeness centrality value of 75 /1.10. But it is worth to mention that it has least control over the other factors interaction as the number of it shortest paths notably small in comparison to other central factors which could be the result of it closeness to more central factors who could provide a shortest path to the other factors. Taken in consideration that it is not connected to many high weighted factors as its eigenvector value is small in comparison to other central factors with a value of 0.78.

Understanding problems of customers is considered the fifth central factor in the risk and success criteria network in relation to its direct interaction with other factors. In contrast, it is not in the top five central factors in terms of its shortest paths between factors and its ability to control the interaction between risk and success criteria; this is supported by its betweenness centrality value of 22.53 as well as its distance to other factors in the network is questionable as it not in the top five factors, which is supported by its higher closeness value of 1.16. Understanding problems of customers is connected to many important risk and success factors in terms of their weight in this network.

Government factors, market demand obsolete, lack of adequate security technologies (e.g., firewalls, encryption, etc.), insufficient consideration of security and safety and quality are considered the most central nodes in terms of their connections to highly weighted factors in the network.

Table 0-5 Top central factors' analyses for Risk factors influencing success criteria

Factor id	Factor names	Degree centrality	Closeness	Betweenness	Eigenvector
R17PR	Resource insufficiency	84.00	1.03	42.04	0.94
R14DEV	Inefficient team capability	81.00	1.06	47.52	0.88
R11DEV	Unrealistic resource planning	81.00	1.06	31.88	0.90
R21ENG	Understanding problems of developers	75.00	1.10	18.57	0.78
R20ENG	Understanding problems of customers	73.00	1.16	22.53	0.87
R16PR	Unrealistic budget	72.00	1.13	24.85	0.73
R12DEV	Communication gaps	65.00	1.21	27.34	0.59
R24PR	Government factors	20.00	0.00	0.00	1.00
R30PR	Market demand obsolete	20.00	0.00	0.00	1.00
R49ENG	Lack of adequate security technologies (e.g., firewalls, encryption, etc.)	20.00	0.00	0.00	1.00
R58ENG	Insufficient consideration of security and safety	20.00	0.00	0.00	1.00
Risk-to-quality	Quality	19.00	0.00	0.00	0.97

8.5.3 *Top central factors' analyses for Success factors influencing success criteria*

From Table 8-6 factors S4DEV, S7DEV, S10ENG, S11DEV, S13DEV, S14DEV, S16DEV, S17DEV, S18DEV, S19PR, S20DEV, S22ENG, S23PR, S24DEV, S27PR, S31DEV, S32DEV, S33DEV, S36PR and S37DEV are the most central factors in terms of degree as they are directly connected to the largest number of factors. Thus, they are considered to be central factors in terms of direct impact on other risk factors or, in other words, their impact can spread directly to the majority of network success factors and success criteria. It should also be noted that these factors are in close proximity to the rest of the risk factors; this is supported by their closeness centrality values, which are 1. This reflects the spread time and speed of their impact on the other factors. In addition, these factors are also the most central factors in terms of their control of other risk and success factors' interactions with and impacts on each other This is supported by their betweenness value of 1.21, which makes them the most central factors in this network. This research has noted that these factors have scored the same value in each centrality value, and believes that, if they were isolated, this would create a mesh topology. In a mesh topology, as has been mentioned in Chapter 5, all factors are connected to each other, which could be due to a number of reasons. The first is the factors' criticality to the software project in terms of their influence on the success criteria. This point agrees with and is supported by Nasir and Sahibuddin's (2011b) research about the criticality of success in software projects, where all the factors above have been found to be critical in both studies. The only factor that was not identified in Nasir and Sahibuddin's (2011b) research is S23PR Pilot project performance. The results of the current study show that this factor has a major influence on the success criteria, due to its important role in building trust with clients (Mishra and Mishra, 2011, Babar et al., 2007).

Table 0-6 Top central factors' analyses for Success factors influencing success criteria

Factor id	Factor names	Degree centrality	Closeness	Betweenness	Eigenvector
S4DEV	Efficient project management	65	1.00	1.21	0.96
S7DEV	Effective communication and feedback	65	1.00	1.21	0.96
S10ENG	Familiarity with technology	65	1.00	1.21	0.96
S11DEV	Appropriate development processes	65	1.00	1.21	0.96
S13DEV	Up-to-date progress reporting	65	1.00	1.21	0.96
S14DEV	Effective monitoring and control	65	1.00	1.21	0.96
S16DEV	Good leadership	65	1.00	1.21	0.96
S17DEV	Risk management	65	1.00	1.21	0.96
S18DEV	Change management	65	1.00	1.21	0.96
S19PR	Appropriate infrastructure	65	1.00	1.21	0.96
S20DEV	Committed and motivated team	65	1.00	1.21	0.96
S22ENG	Managing omplexity, project size, number of organisations involved	65	1.00	1.21	0.96
S23PR	Pilot project performance	65	1.00	1.21	0.96
S24DEV	Clear assignment of roles and Responsibilities	65	1	1.21	0.96

8.6 Summary

This chapter has revealed the general characteristics for three networks by using statistical analysis of several measures of the interaction and network centrality. Additionally, the chapter has identified the most central factors in each network and explored their impact.

Chapter 9: Interdependency between risk and success

Chapter 9: Interdependency between risk and success

9.1 Introduction

In order to know and analyse the dependency and influence between the risk and success factors in software development projects, a network of interactions that occur between them has been created through the use of a dependency matrix. This chapter studies the interactions of the four criteria in terms of their degree and closeness centrality measures. First, the general network characteristics and findings are presented. Then the top five factors of the network are identified, analysed and explained, and statistical comparisons are also presented. The interdependency of each criterion will be analysed in chapters 10, 11, 12 and 13.

9.2 Software development project interaction network

The network in Figure 9-1 shows the interactions between all the four criteria, cost, quality, time and scope with the risk and success factors. In this network, the ego network topology has been applied to each and every criterion in the network to identify the factors that are connected to all the other factors as well as to study the centrality measures between all the four criteria. The results show that 80 factors are directly connect to cost, quality, time and scope out of 106 possible factors. This means that there are 80 risk and success factors that have a positive or a negative influence on software project development , whilst 26 factors are not directly connected to the four criteria. In this network, as has been mentioned, the ego network has been applied to all the criteria, which in turn will show that all the criteria have the same degree score, as the idea is to study the interaction of the factors connected to all criteria. Also, as a result of this, the closeness centrality of all the criteria has a centrality of 1. Thus, the main focus will be on what the top five factors are in terms of their centrality to the success criteria.

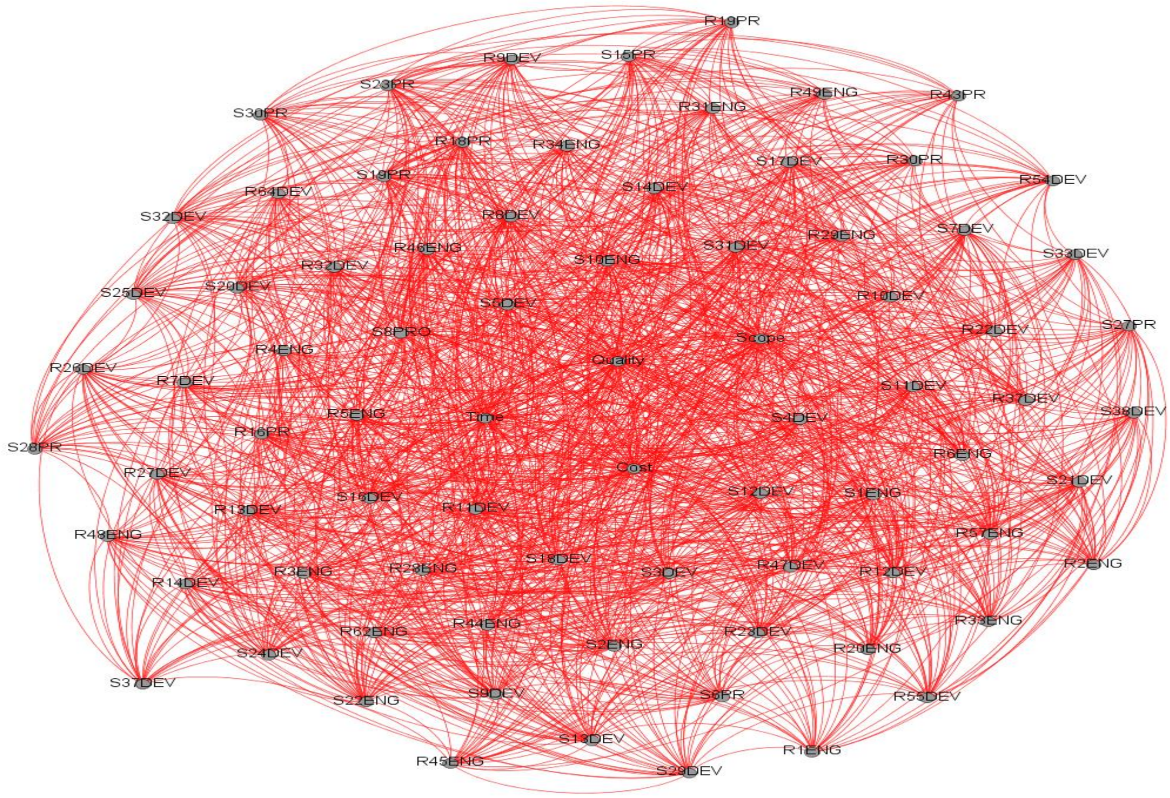


Figure 9-1 Software development project interaction network

9.2.1 *Top five Success factors influencing the success criteria*

The top five factors that are central to all factors in terms of the degree are S4DEV Effective project management with a degree of 46 interactions, followed by S31DEV, S14DEV and S8PR with a degree of 45. Although four factors have a degree of 44 (S3DEV, S12DEV, S18DEV and S5DEV), S18DEV is considered to be the 5th central factor due to its high betweenness value of 57.6, compared to the other three factors. In relation to software development, Effective project management is the factor with the most influence on project success. This shows that all stages of a software project are affected by it, and also indicates how important it is that decision-makers consider this factor. In addition, all the top five factors contribute significantly to the success or failure of the project's development from an interaction point of view. An in-depth investigation of their impact will be provided in the next sections.

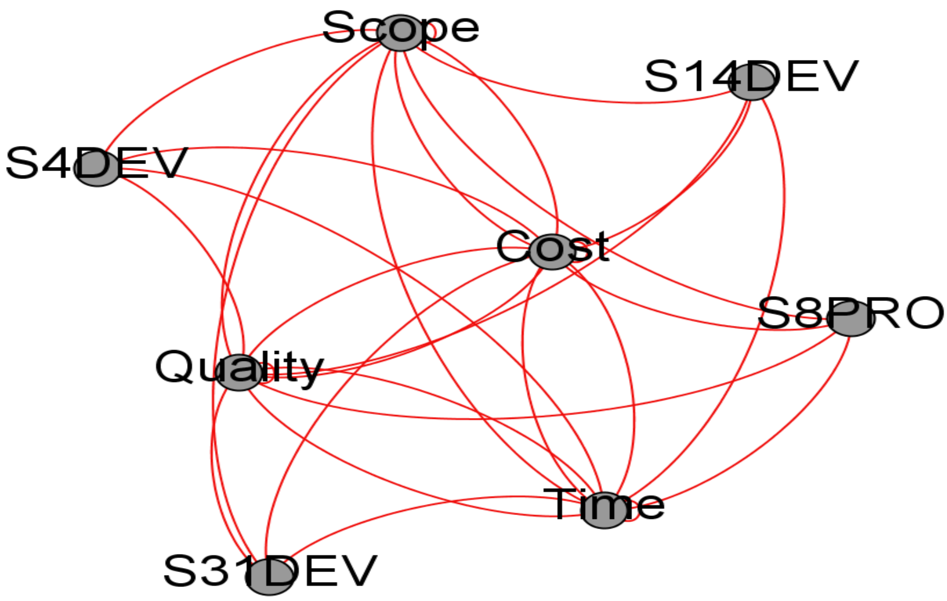


Figure 9-2 Top five central factors to the success criteria

9.2.2 *The centrality impact of Success factors influencing the success criteria*

In order to identify the influence and importance of the most central factors on the software project criteria, two tables have been produced. Table 9-1 lists the top five central factors; the degree of each factor is analysed in terms of its value in relation to each criterion as well as its overall value. Furthermore, the ranking of each factor in the network is listed as well as the overall ranking depending on the average degree value of the criteria score. lists the percentages for those results in order to compare them to the overall nodes available.

Table 9-1 Degree's top five central factors

Factor	Cost	Quality	Time	Scope	Statistically overall average degree	Average ranking	Degree centrality in all criteria network	Betweenness	Ranking in network
S4DEV	63	69	62	53	61.75	1	46	63.76	1
S8PR	62	66	60	50	59.5	3	45	60.57	2
S14DEV	60	65	58	51	58.5	4	45	60.36	3
S31DEV	61	67	60	52	60	2	45	60.35	4
S18DEV	57	63	57	51	57	5	44	57.61	5
Criteria average score	60.6	66	59.4	51.4					

Table 9-2 The percentages of connections to the total number of connectionsTable 9-3 The percentages of connections to the total number of connections

Factor	Cost	Quality	Time	Scope	Overall average degree	Average ranking	Degree centrality in all criteria network	Ranking in network
S4DEV	59.43%	65.09%0	58.49%	50.00%	58.25%	1	43.40%	1
S8PR	58.49%	62.26%	56.60%	47.17%	56.13%	3	42.45%	2
S14DEV	56.60%	61.32%	54.72%	48.11%	55.19%	4	42.45%	3
S31DEV	57.55%	63.21%	56.60%	49.06%	56.60%	2	42.45%	4
S18DEV	53.77%	59.43%	53.77%	48.11%	53.77%	5	41.51%	5
Criteria average score	57.17%	62.26%	56.04%	48.49%				

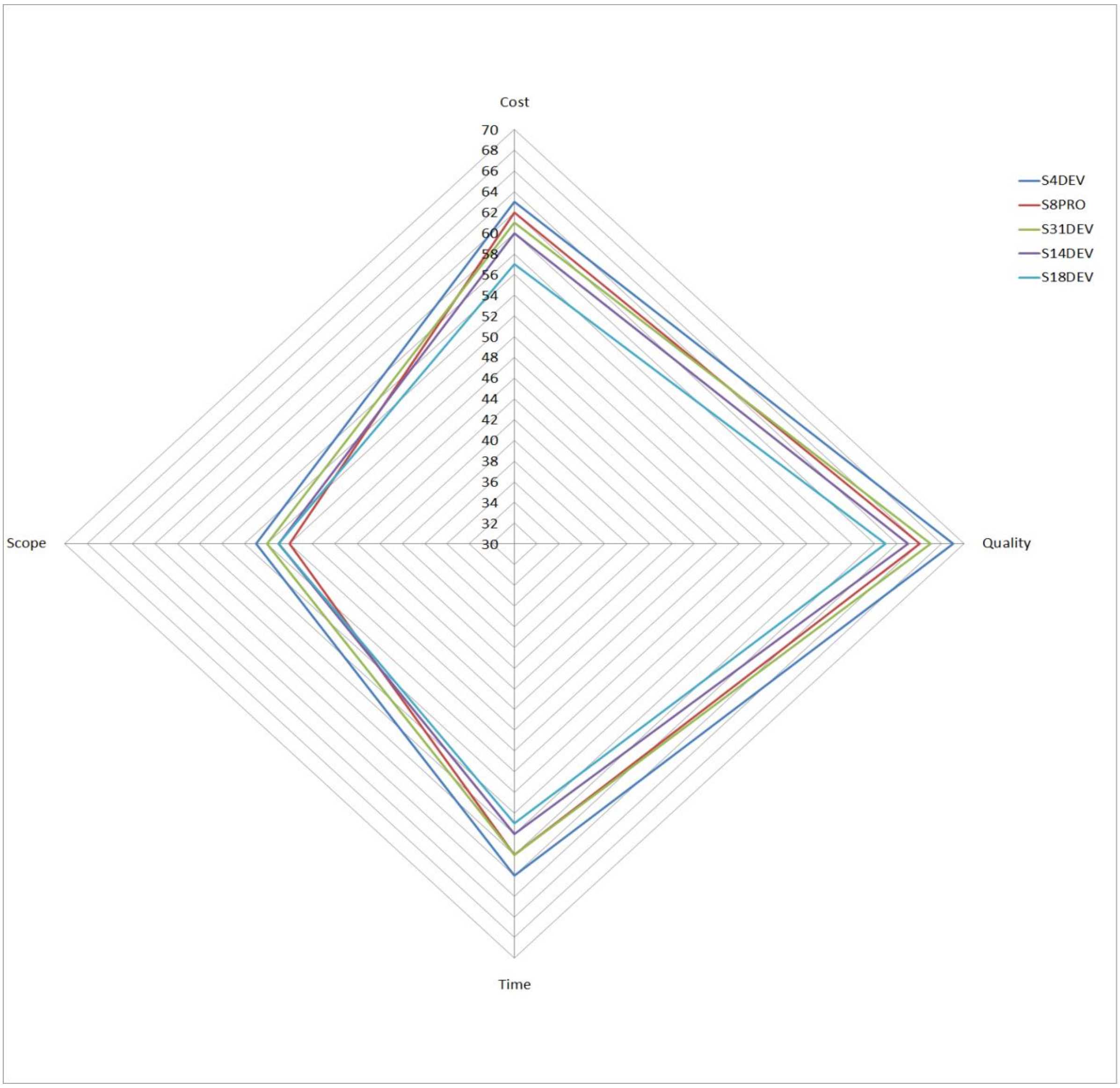


Figure 9-3 Degree's top five central factors

From the tables, it is obvious that S4DEV is the most central factor to all criteria as it has been ranked as the top central factor in the statistically average ranking and it is ranking in the network. Furthermore, S4DEV is connected to 43.40% of the 80 factors connected to all criteria. It also has an average connection of 58.25% of the factors to each criterion. Quality will be the criterion most affected by this factor, due to the large number of ego network connections. Cost is affected slightly more than time by this factor as it has 59 interactions compared to 58 interactions for time. Effective project management has the lowest interaction in the scope ego network as a criterion in the software development project; this is due to the low number of interactions in the scope ego network. That does not mean that this factor has a low impact on scope; it is absolutely the opposite of that, as it is the most central factor in the scope ego network.

The Table 9-1 results show that there are significant differences between the S8PR ranking in the all criteria network, where it is ranked as the second most central factor with degree of 45 and betweenness value of 60. Additionally, S14DEV Effective monitoring and control has connections to 42.45% of the total nodes available in this network. On the other hand, S14DEV is ranked as the third central factor in the average connection to each criterion with an average degree of 59 connections. Furthermore, it has an average connection of 56.13% to all criteria. The reason for this significant difference is that S4DEV is ranked as second in the cost, quality and time networks with degrees of 58.49%, 62.26% and 56.60% respectively of the total nodes available, but it is ranked third in the scope ego network with a connection to 47.17% of the total number of nodes.

Although S14DEV has the same degree value as S8PR, of 45, and it has connections to 42.45% of all nodes in the network, it has a lower betweenness value of 60.36, which makes it the third central factor in the all criteria network. In comparison, S14DEV is ranked as the fourth central factor in terms of its statistically average connections to all criteria as it has an average connection of 58.5 or 55.19% connections to all nodes in the network. The main reason behind that it has ranked as fourth central factor in all the four ego network criteria.

In terms of its degree centrality, S31DEV Organisational culture is ranked as the fourth central factor in the network as it has a betweenness value of 60.35. S31DEV has the same degree as S8PR and S14DEV. In spite of its statistically average connections to all criteria, S31DEV is ranked as the second most central with a degree of 60 and has on average, a 49.06% connection with all nodes available in each criterion'. S31DEV is considered the second central factor to quality, time and scope, whilst it ranked third in terms of degree centrality to cost. From the chart above, it can be seen that this factor has 63.21% of the total connections available in the quality ego network, 56.60% of the total connections available in the time ego network, and 49.06% of the total connections available in the scope ego network. S31DEV is ranked as the third most central factor in the cost ego network as it has a degree of 61.

S18DEV is considered the fifth most central factor in the all criteria network with a degree of 44 and it has connections to 41.51% of the nodes in this network. This result is also confirmed by its betweenness centrality value of 57.61. Similarly, S18DEV has ranked fifth in the statistically average connections to all criteria with an average degree of 57. Likewise, it has an average connection of 48.11% to each criterion. In quality, time and cost S18DEV is ranked as the fifth central node, whilst in scope it is ranked as the fourth, as it connected to 48.11% of the nodes available in the scope ego network.

In terms of their influence on software development, these results mean that that S4DEV is the factor that, if it is applied in a project, it will have a positive influence on 43.40% of the factors and criteria. Furthermore, the results show that this factor is a very important factor that should be used to increase the chances of project success as it should be the main factor that any software project should have. In addition, realistic budget, effective monitoring and control, organisational culture and change management are factors that decision-makers should build their planes and strategies around.

9.2.3 *Top five risk factors influencing the Success criteria*

Table 9-4 Degree's top five risk factors to all criteria

Factor	Cost	Quality	Time	Scope	Statistically Overall average degree	Average ranking	Degree centrality In all criteria network	Ranking In network	Betweenness
R16PR	41	41	41	41	41	1	37	1	33.02
R23DEV	40	40	40	40	40	2	36	2	31.25
R11DEV	40	39	39	39	39.25	3	36	3	31.08
R18PR	38	38	39	38	38.25	5	35	4	28.97
R3ENG	39	39	39	39	39	4	35	5	28.67
Criteria average score	39.6	39.4	39.6	39.4					

Table 9-5 The percentages of connections to the total number of connections

Factor	Cost	Quality	Time	Scope	Overall average degree	Average ranking	Degree centrality In all criteria network	Ranking In network
R16PR	38.68%	38.68%	38.68%	38.68%	38.68%	1	34.91%	1
R23DEV	37.74%	37.74%	37.74%	37.74%	37.74%	2	33.96%	2
R11DEV	37.74%	36.79%	36.79%	36.79%	37.03%	3	33.96%	3
R18PR	35.85%	35.85%	36.79%	35.85%	36.08%	5	33.02%	4
R3ENG	36.79%	36.79%	36.79%	36.79%	36.79%	4	33.02%	5
Criteria average score	37.36	37.17	37.36	37.17				

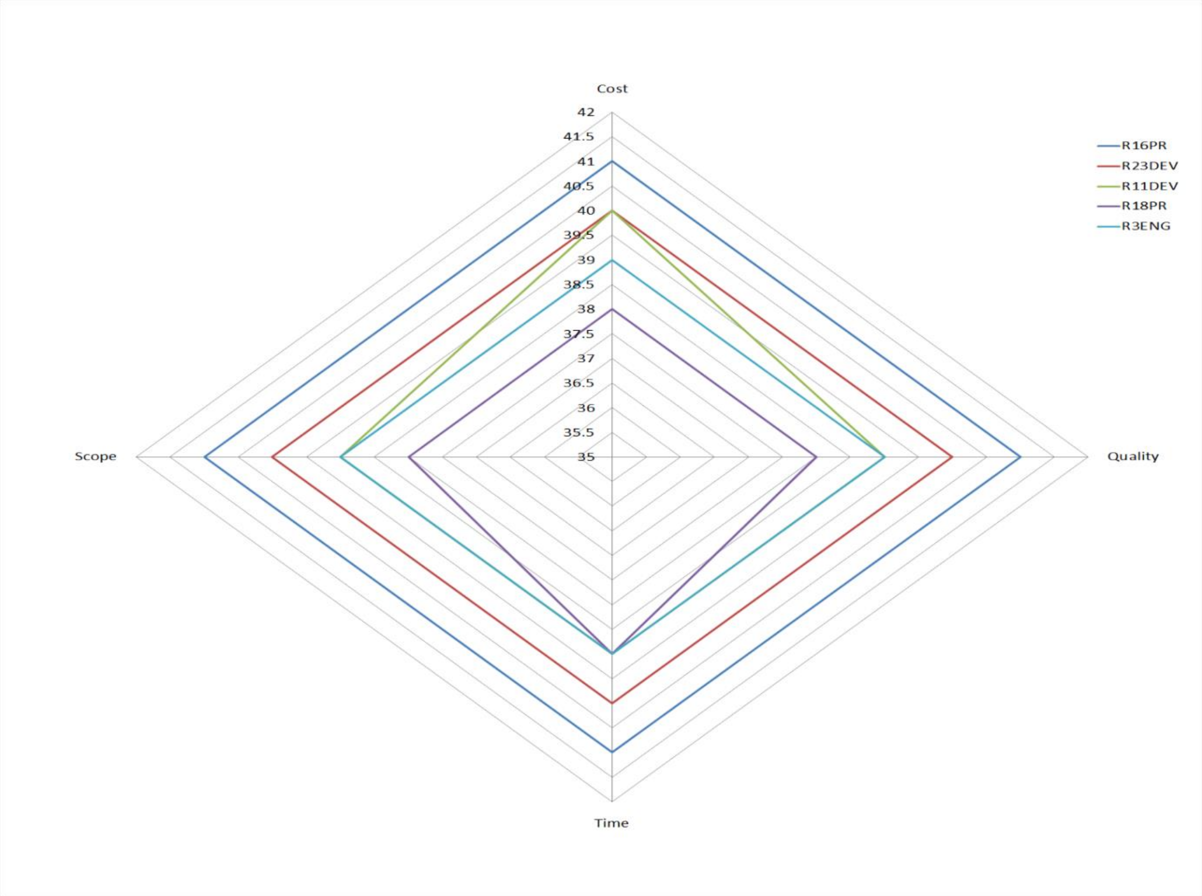


Figure 9-4 Degree's top five risk factors to all criteria

R16PR has been ranked as the most central factor both statistically and in the network of all criteria. It has a connection to 37 nodes in the network. Also, it is interesting to see that R16PR has the same number of connections to each criterion as all the ego criteria, as it has exactly the same degree value of 41. It also has a 38.68% connection to each criterion. This puts an emphases on the importance of this factor in all four criteria in software projects, with a betweenness value of 33.06.

Similarly, R23DEV Project manager lacks experience has ranked the same in the network as well as in the statistically average connections to each criterion, as it has been ranked as the second most central node in the network. Furthermore, R23DEV has a degree value of 36 in the network with connections to 33.96% of nodes available. In addition, Project manager lacks experience has the same connections of 37.74% to each criterion. It is also worth mentioning that, although each criterion has a degree of 40, they are not connected to the same factors; there are slightly different factor connections.

The third central factor in terms of degree centrality in the network as well as in the statistically average connections to all criteria is R11DEV Unrealistic resource planning. Although its degree centrality in the network is 36, which is the same as R23DEV, due to its betweenness value of 31.08, which is less than R23DEV, it is ranked as the third most central risk factor in the network. It is noticeable that Unrealistic resource planning has more connections and effect in the cost ego central network with a degree of 40 than the quality, time and scope ego networks.

R18PR User resistance has significantly different ranking between the network and the statistically average connections to each criterion. On the statistically average connections to each criteria it has ranked as the fifth most central factor in the network as it has an average connections of 38.25%. Furthermore, it is noticeable that time is considered to be the criterion that is most criteria affected by this factor, with a degree of 39 connections. On the other hand, user resistance is ranked as the fourth most central factor due to a combined measurement value degree of 35 and betweenness centrality value of 28.97.

In the network of all criteria, R3ENG Lack of technical skills has been identified as the fifth most central factor, although it shares the same degree centrality as R18PR; however, it has a lower betweenness centrality value of 28.67 than R18PR. Also, this factor has connections to 33.02% of all nodes in the network. In addition, Lack of technical skills is ranked as the fourth most central factor in the statistically average connections to each criterion as it has average connections of 36.79% to each criterion. Furthermore, it has a degree of 39 to each criterion. It is worth mentioning that R3ENG has the same degree as R11DEV in the ego networks of quality, time and scope.

Above all, these results show that the top five factors have almost the same impact influence on software projects. Therefore, decision-makers should emphasise the need for a plan to avoid having “Unrealistic budget” during the requirements stage as, if this risk occurs, it will have an impact of 34.91%. The other four factors have a slightly similar impact, and their influence on software projects will be discussed in more detail in Chapter 14

9.3 Summary

This chapter has analysed the relationship between the four criteria in the network. In addition, it has uncovered the most important factors in terms of their centrality in the network and has also compared their centrality results.

Chapter 10: Cost Ego network

Chapter 10: Cost Ego network

10.1 Introduction

The ego network has become a standard unit of measurement for studying small-scale interactions. This chapter analyses the interdependency in the cost ego network, identifying the top centrality risk and success factors, and then identifying and isolating the most controlling risk and success factors. After that, the contribution of the isolated network factors in cost prediction is analysed. The chapter ends by analysing the association between risk and success factors in the ego network.

10.2 Cost ego network

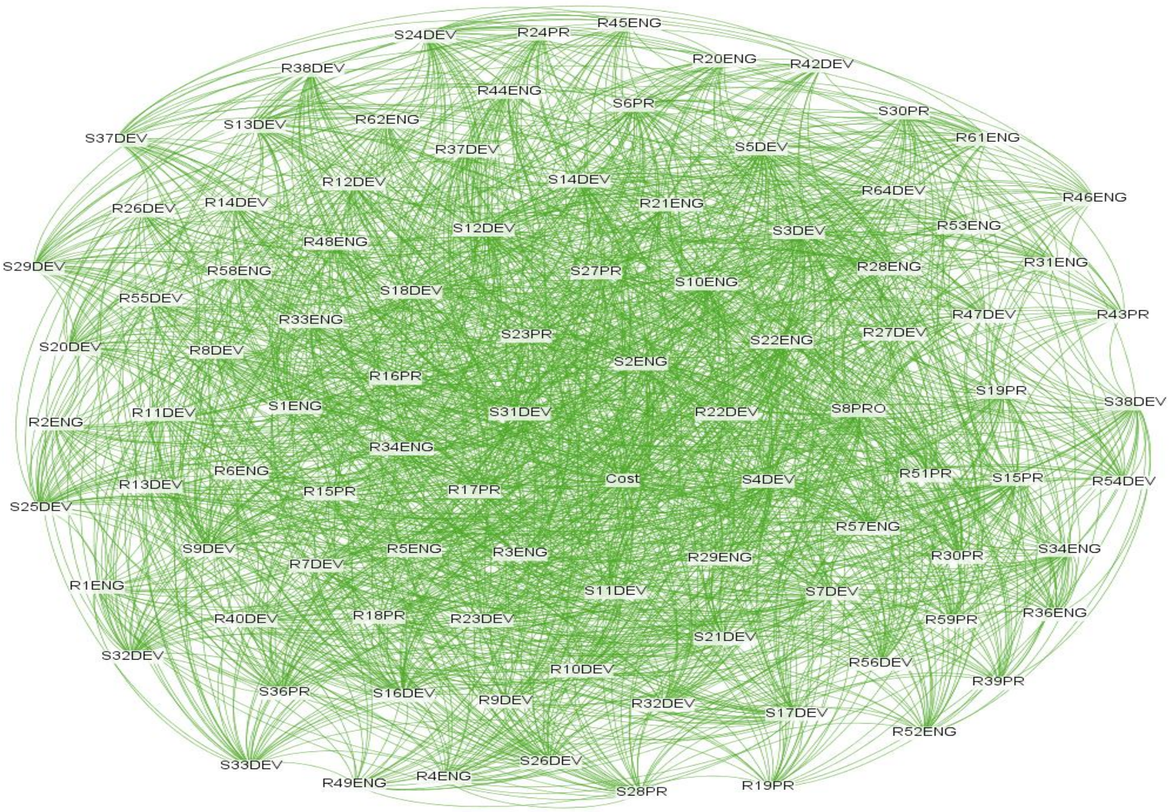


Figure 10-1 Cost ego network

In this network, there are 96 factors directly connected to cost out of 102 possible factors. Six factors and criteria are not directly connected to cost. The other three criteria (quality, time and scope) have been isolated and removed from the network to study the interaction of the risk and success factors in relation to cost without the influence of the other success criteria, to obtain the maximum interaction impact on the cost itself.

The centrality measures are the focus of the cost ego network degree, as it shows the number of interactions in the network. Closeness shows how close a factor is to every other factor in the network. Betweenness centrality has been used to find the factor that has the shortest paths and controls the interaction in this ego network later in this section.

10.2.1 *Top five central success factors*

Figure 10-2 Cost ego network with a degree of 59

S4DEV Efficient project management is the most central factor in terms of degree to the cost ego network with a degree of 59 connections as it is directly connected to the largest number of factors. Thus, this factor is considered the central factor in terms of direct impact on cost or, in other words, its impact can spread directly to the majority of the network's risk and success factors. A more sophisticated centrality measure is closeness, which emphasises the distance of a node from all others in the network by focusing on the geodesic distance from each node to all others (Umadevi, 2013). Furthermore, considering that the more central a node is the lower its total distance from all other nodes, S4DEV is the most central factor in terms of closeness to the cost ego network. It should also be noted that this factor is in close proximity to the rest of the risk factors in the cost ego network; this is supported by its closeness centrality value of 1.38, which reflects the spread time and speed of this factor's impact on the other factors. Khan et al. (2009) also found that efficient project management was an important factor in cost saving in Asia, which supports the results of this research.

Figure 10-3 Cost ego network with a degree of 58

S8PR Realistic budget is the second factor in terms of its impact on cost as it has degree of 58 connections. This emphasises the importance of cost saving. According to Bassett et al. (2010), "One of the most important aspects of planning your project is your budget. Whether you are embarking on a large- or small-scale project, there will be associated costs", and "By taking the time to plan for these things in advance, you can get a good lead on anticipating cost-associated risks so that they don't take over your project". Likewise, it also has a closeness value of 1.39, which is slightly higher than S4DEV, which shows that it is the second factor in terms of its time and speed impact on the other risk factors. According to Akbar et al. (2012), "Best project management practices used by project managers can optimise software project cost estimation and management".

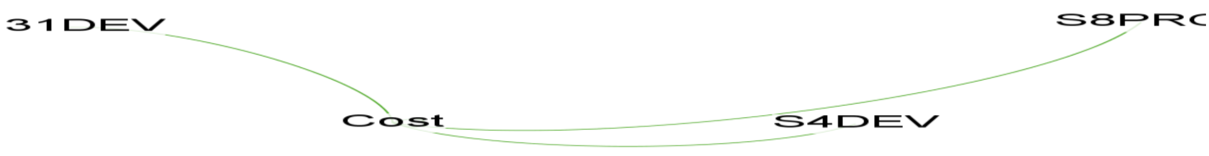


Figure 10-4 Cost ego network with a degree of 57

S31DEV Organisational culture is the third factor in terms of its influence on Cost, as it has the third shortest distance to other risk factors in this ego network. S31DEV has a direct connections with 57 risk factors. According to Atkinson et al. (2006), “This culture can be manifest in a variety of ways, such as attitude to: planning, formal procedures, regulations, criticism, mistakes, uncertainty, and risk”. All those project characteristics have an effect on the project cost. This shows the centrality and the importance of organisational culture on the cost ego network as it has the ability to affect the project risk factors. Above all, the closeness of organisational culture has the fastest spread time to other risk factors, and the degree shows the number of factors directly affected by this factor.

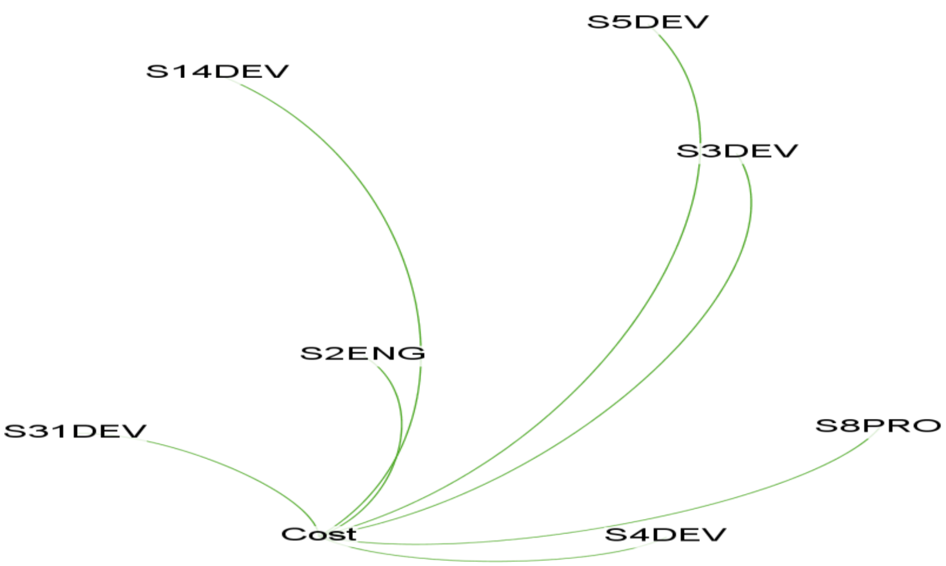


Figure 10-5 Cost ego network with a degree of 56

The fourth central factor is S2ENG Clear objectives and goals, which emphasises the importance of the planning stage as it has a big effect on the project cost. Clear objectives and goals have a degree of 56 connections. Also, the factor has a closeness value of 1.41, which is a slightly lower centrality than Organisational culture. Although S3DEV, S14DEV and S5DEV have the same centrality as S2ENG in terms of degree centrality, S2ENG is the most central factor in terms of its betweenness value of 117.48, which means that this factor has more shortest paths than S3DEV, S14DEV and S5DEV. Furthermore, the extensive literature search revealed that most practitioners consider clear objectives and goals to be one of the most critical success factors that contribute to project success (Schmidt et al., 2001; Keil et al., 2002; Taylor, 2006, Standish Group, 2006, 2009 and Nasir, & Sahibuddin, 2011). Where cost as criteria have been important the project success. Clear objectives and goals have the ability to give the decision-makers the motivation and the platform to choose the right solution to any situation that occurs during the project life cycle (Jamieson and Hyland, 2006). This type of clarity will affect other factors in a software project and will also positively affect cost saving in the project.

As has been mentioned, S2ENG, S3DEV, S14DEV and S5DEV have the same degree and closeness centrality but S3DEV Realistic scheduling is considered to be more central than S14DEV and S5DEV as it has the highest number of shortest paths as well as the advantage of controlling the indirect connections between factors in the cost ego network after S2ENG, with a betweenness vale of 117.00. The importance of this factor is heavily based on its effect on the planning stage. Therefore, the project cost and scheduling are developed during this stage. A realistic schedule is important as many projects fail in delivering the product on time, as well as being over-budget, due to unrealistic and optimistic schedules and expectations from project stockholders (Saini and Hooda, 2014).

10.2.2 *Top five central Risk factors in Cost ego network*

In order to identify the top risk factors, this research uses the centrality data from the cost ego network. Only the risk factors have been analysed and compared. Furthermore, the top central factors have been identified, as can be seen in Table 10-1.

Table 10-1 Top five central Risk factors in Cost ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
R16PR	38	1.60	35.50708	0.454912	1
R17PR	37	1.60	33.13701	0.445469	2
R11DEV	36	1.62	31.37436	0.433415	3
R23DEV	36	1.62	31.22434	0.434258	4
R3ENG	35	1.63	28.91228	0.425401	5

Although R16PR Unrealistic budget and R17PR Resource insufficiency have the same degree value of 37 and closeness value 1.6, R16PR has a different betweenness value of 35.51, which means it has more control of factors interacting with each other in this ego network. Baccarini et al. (2004) ranked unrealistic budget and schedule as the second highest risks to IT projects. Also, the importance of this factor in terms of its effect on other factors is based on its occurrence during the requirement analysis (Quadri et al., 2015).

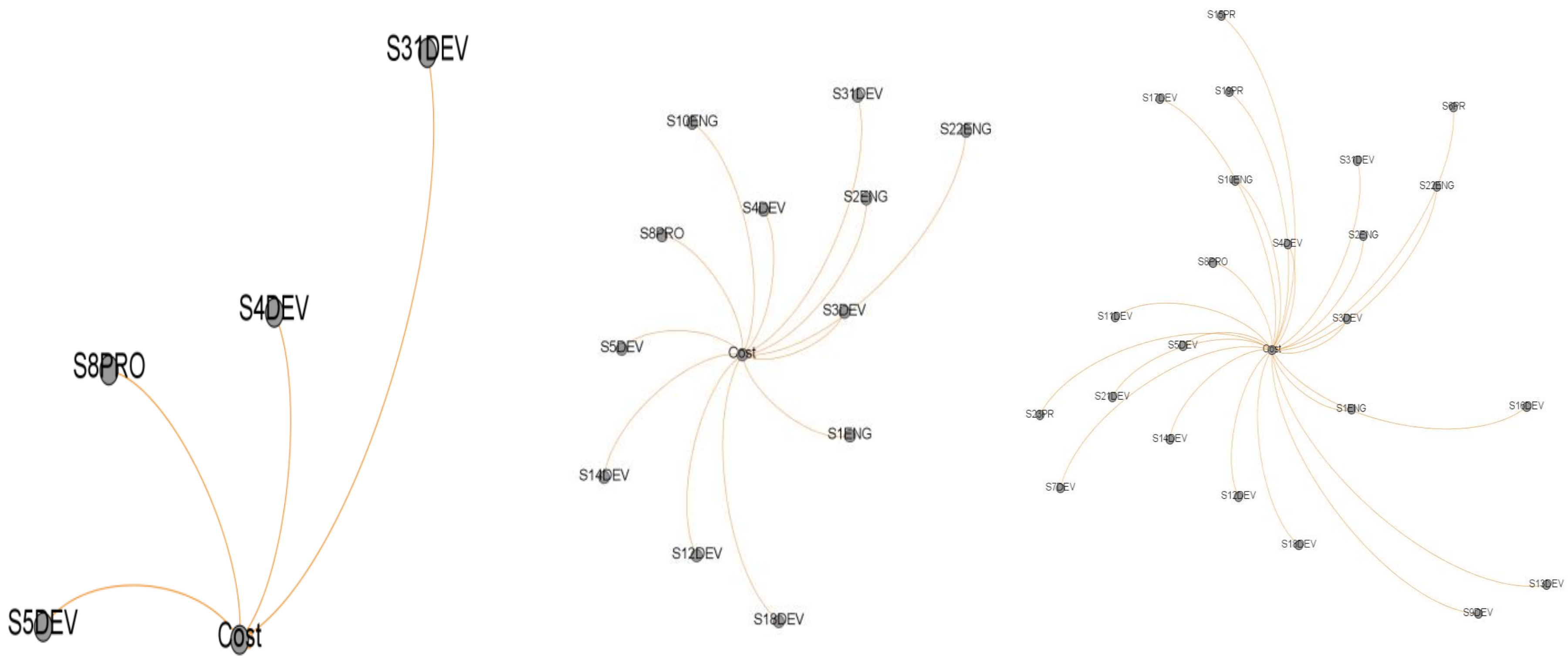
Resource insufficiency is considered as the second most central factor to the cost ego network as it has a betweenness value of 33.14. Additionally, one of the aspects of resource inefficiency is that it can play a detriment role on the project size (Koh, 2011). According to Mendes (2007), resources are essential to a software project’s schedule preparation. Also, its main risk effect lies in project managers finding it difficult to control any resource insufficiency during the project’s implementation, which in turn will have an effect on other factors (Hung et al., 2013).

R11DEV Unrealistic resource planning and R23DEV L Project manager lacks experience are the third and fourth centrality risk factors. Although they have the same number of connections to the other factors with a degree of 36 and are the same distance from all other factors with a closeness of 1.62, R11DEV has more shortest paths and has more control of the factors' interactions with each other with a betweenness value of 31.37 compared to R23DEV's betweenness centrality value of 31.22. This result agrees with Rosencrance (2007) as in her research on over more than 1,000 IT professionals she reports that "Two out of the three most important causes of an IT project failure are perceived to be related to poor effort estimation, in particular insufficient resource planning" (Trendowicz and Jeffery, 2014a).

R3ENG Lack of technical skills is the fifth most central factor with a degree of 35 connections and a closeness centrality value of 1.63. Which shows the influence of lack of technical skills on the other factors as it could affect large number of factors in a very short space of time.

10.3 Isolating the risk and success factors related to the cost ego network

The aim of this section is to find the factor controlling the influence of risk factors over the success factors. A three-step methodology has been applied. First, the ego network algorithm has been applied to isolate the success factors related to cost. Then the degree range algorithm has been applied to determine and give an overview of the graph centrality factors in terms of their degree as well as to explore the dynamics of the graph. Then the success factors with high betweenness value have been selected as well as the risk factors associated with them in the cost ego network. The same steps have been applied to find the factor controlling the influence of success factors over the risk factors.

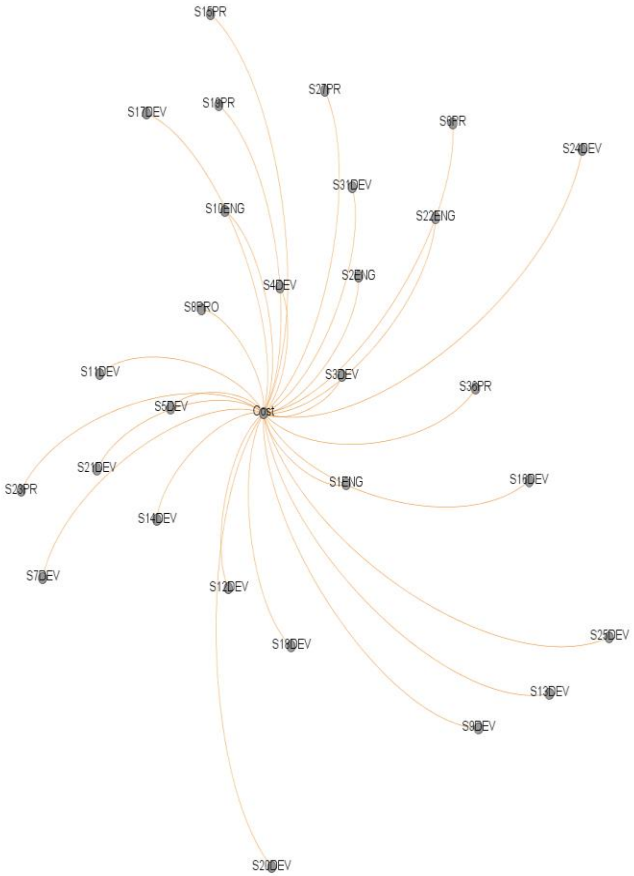


This graph shows the success factors with a degree of 62 and more in the Cost Ego network

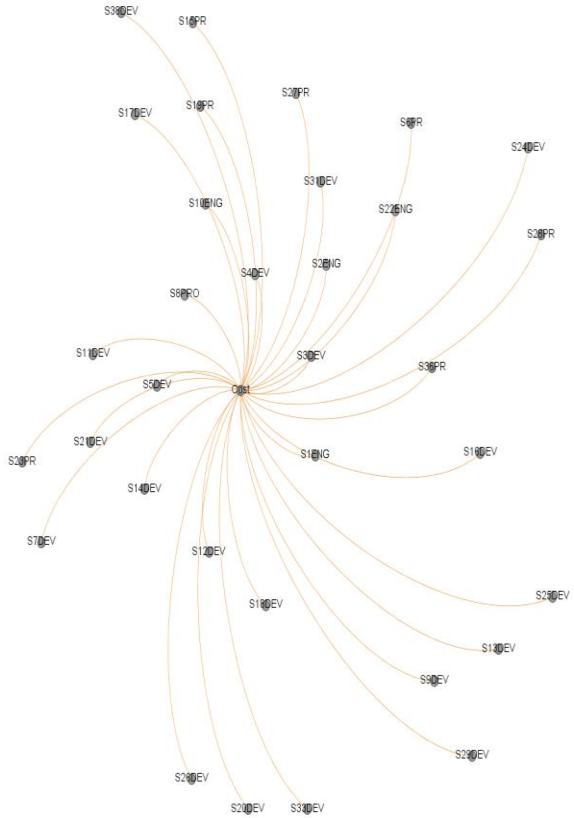
This graph shows the success factors with a degree of 57 and more in the Cost Ego network

This graph shows the success factors with a degree of 52 and more in the Cost Ego network

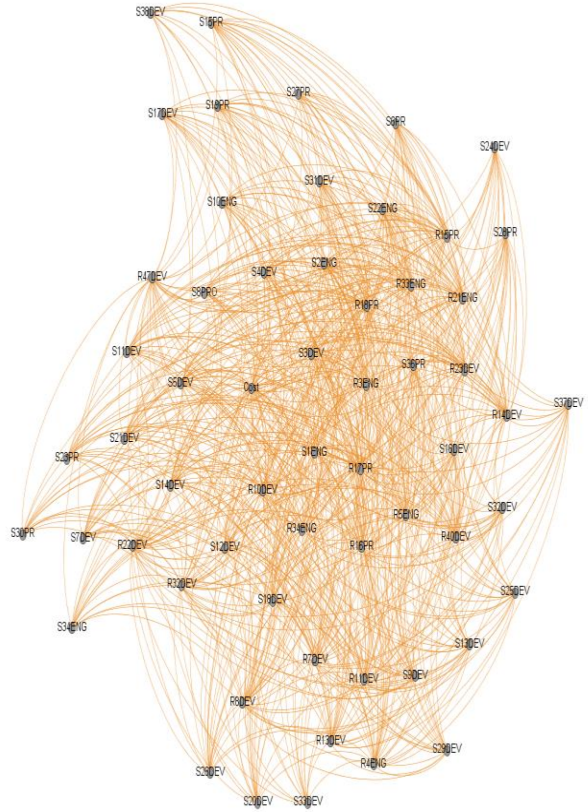
Figure 10-6 The change that occurs in the quality ego network and its success factors when adjusting the degree topology value part



This graph shows the success factors with degree of 47 and more in the Cost Ego network



This graph shows the success factors with degree of 42 and more in the Cost Ego network



This graph shows the success factors with degree of 34 and more in the Cost Ego network

Figure 10-7 The change that occurs in the cost ego network and its success factors when adjusting the degree topology value part two

10.3.1 *Risk factors influencing cost*

The algorithm has been applied in various steps. In the degree range of 62 it is noticeable that there are four most central factors, S4DEV, S5DEV, S8PR and S31DEV. When reducing the minimum degree range to 57, the number of factors connected to cost increases to 12. In the minimum degree range of 52, there are 23 factors connected to the cost ego network. Twenty-eight factors are respectively central to cost with an increase of 21.7% when a degree of 47 is applied. As can be seen above, there are 33 factors with a degree of 42 or higher. The last degree range applied in the cost ego network is 34, and there are 58 factors interacting with cost directly.

10.3.1.1 *The success factor that has the most control and influence over the risk factors in cost ego network*

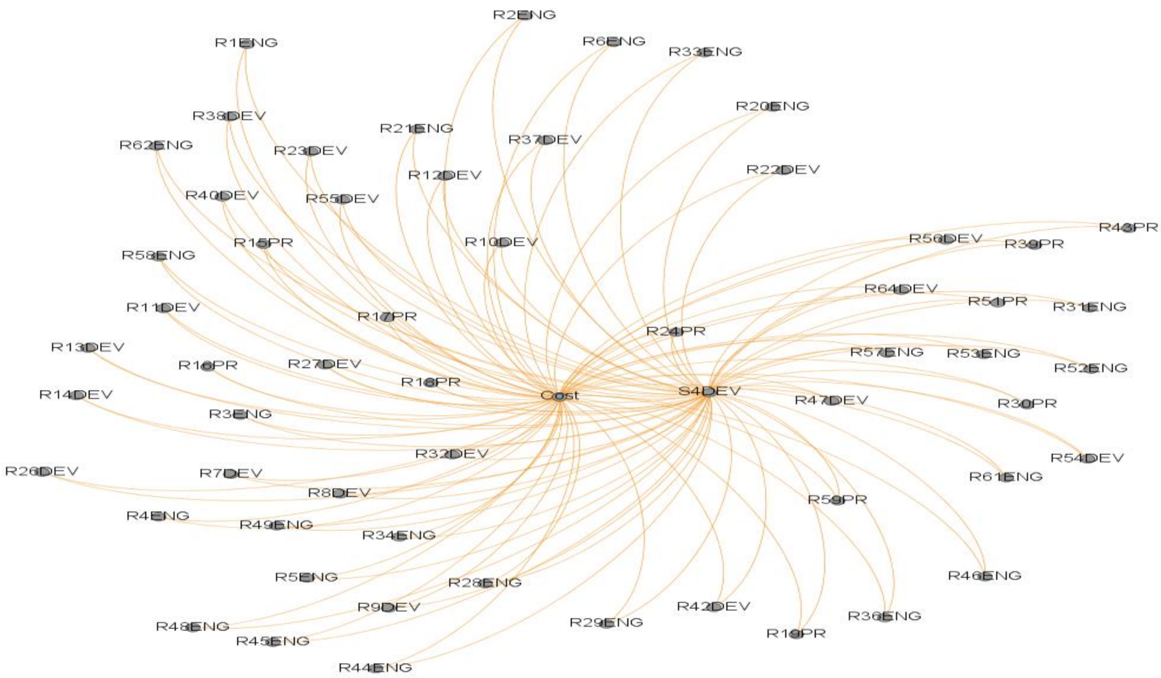
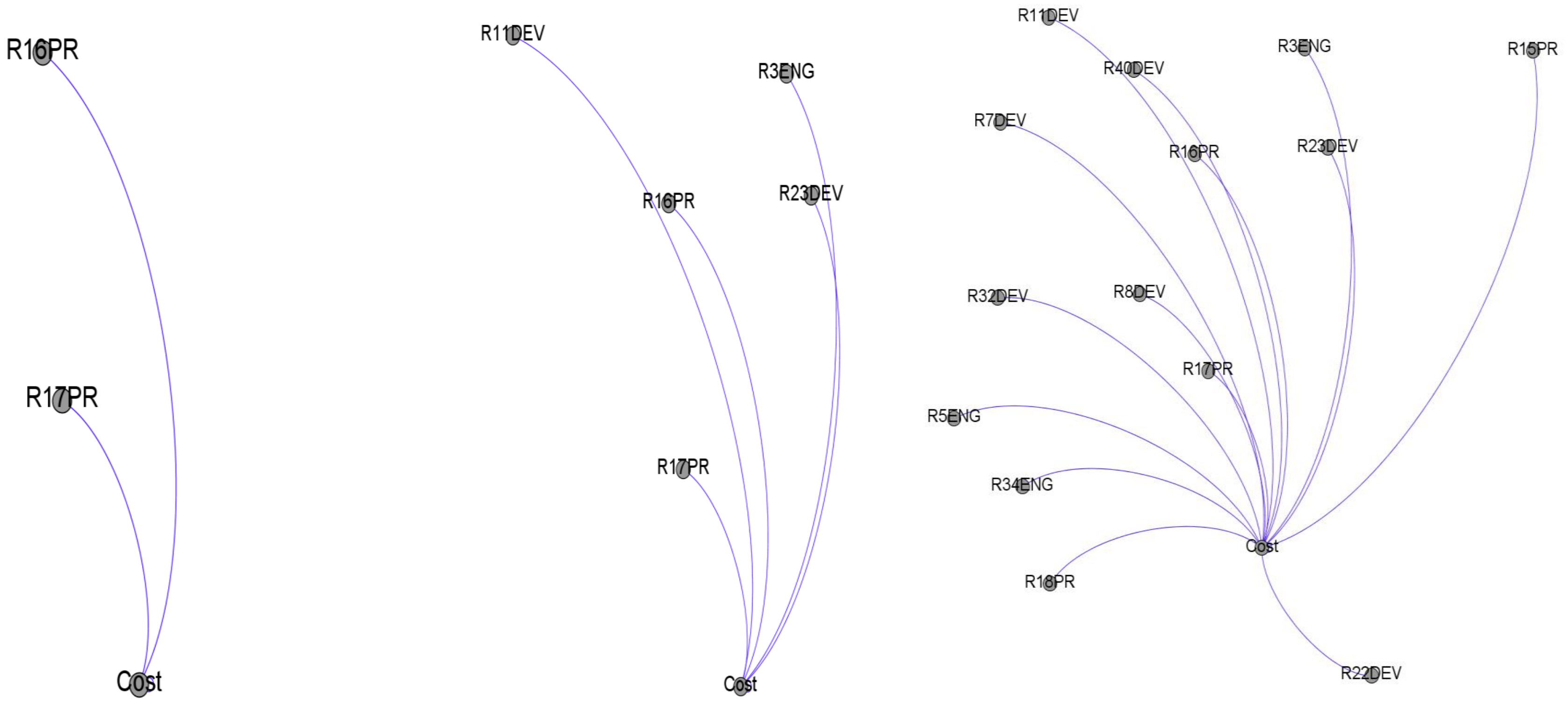


Figure 10-8 S4DEV's influence on risk factors in the cost ego network

It is noticeable that the most central node in terms of its betweenness is S4DEV Effective project management, as has been mentioned in the previous section. Also, the percentage of factors associated and interacting with this factor in the cost ego network are 61% of the total number of connections in this network. S4DEV is also the most central factor in the network in terms of its betweenness value of 131.52. Furthermore, as S4DEV is connected to 58 risk factors connected to the cost ego network, it controls the interaction of about 91% of the risk factors connected to the cost ego network. It should also be noted that this factor is in close proximity to the rest of the risk factors; this is supported by its closeness centrality value of 1.4, which reflects the spread time and speed of this factors' impact on the other factors. Most major project failures are related to social issues. According to Langer et al. (2014), "Prior research on project management has stressed the need for PMs to have skills and experience in communication, leadership, and project-related technology". O'Connor and Laporte (2012) noted that applying effective project management techniques has a huge impact on software project cost saving.

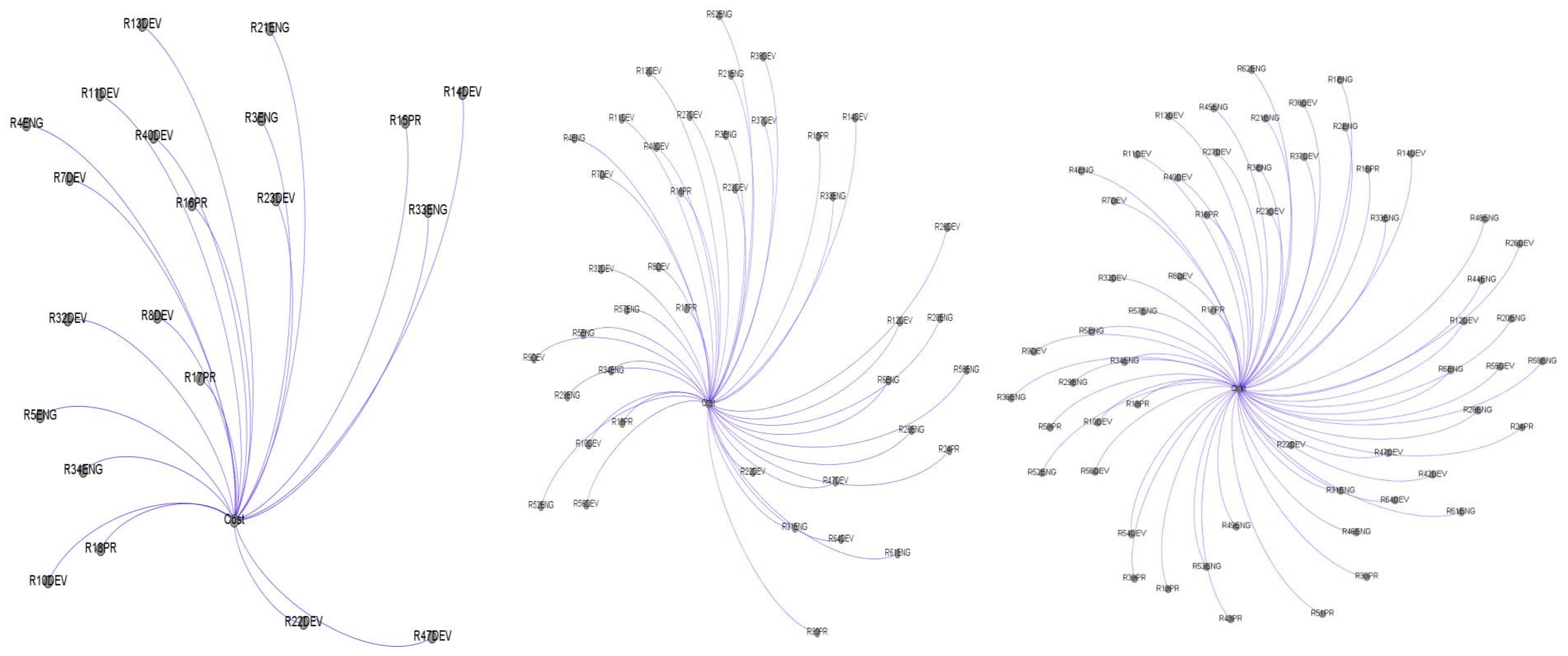


This graph shows the risk factors with a degree of 37 and more in the Cost Ego network

This graph shows the risk factors with a degree of 36 and more in the Cost Ego network

This graph shows the risk factors with a degree of 35 and more in the Cost Ego network

Figure 10-9 The change occurs on cost ego network and its risk factors when adjusting the degree topology value part one



This graph shows the risk factors with a degree of 34 and more in the Cost Ego network

This graph shows the risk factors with a degree of 30 and more in the Cost Ego network

This graph shows the risk factors with a degree of 25 and more in the Cost Ego network

Figure 10-10 The change occurs on cost ego network and its risk factors when adjusting the degree topology value part two

10.3.2 *Success factors influencing cost*

The algorithm has been applied in various steps in the degree range of 17. It can be seen from the graph above that there are two factors connected to cost, R16PR Unrealistic budget and R17PR Resource insufficiency. When reducing the minimum degree range to 36, the number of factors connected to cost increases to six. In the minimum degree range of 35, there are 15 factors connected to the cost ego network whilst 22 factors are respectively connected to cost when the degree of 34 is applied. As can be seen above, there are 42 factors with a degree of 30 or higher. The last degree range applied in the cost ego network is the minimum degree range of 25 with 59 factors interacting with cost directly.

The risk factor that has the most control and influence over the success factors in cost ego network

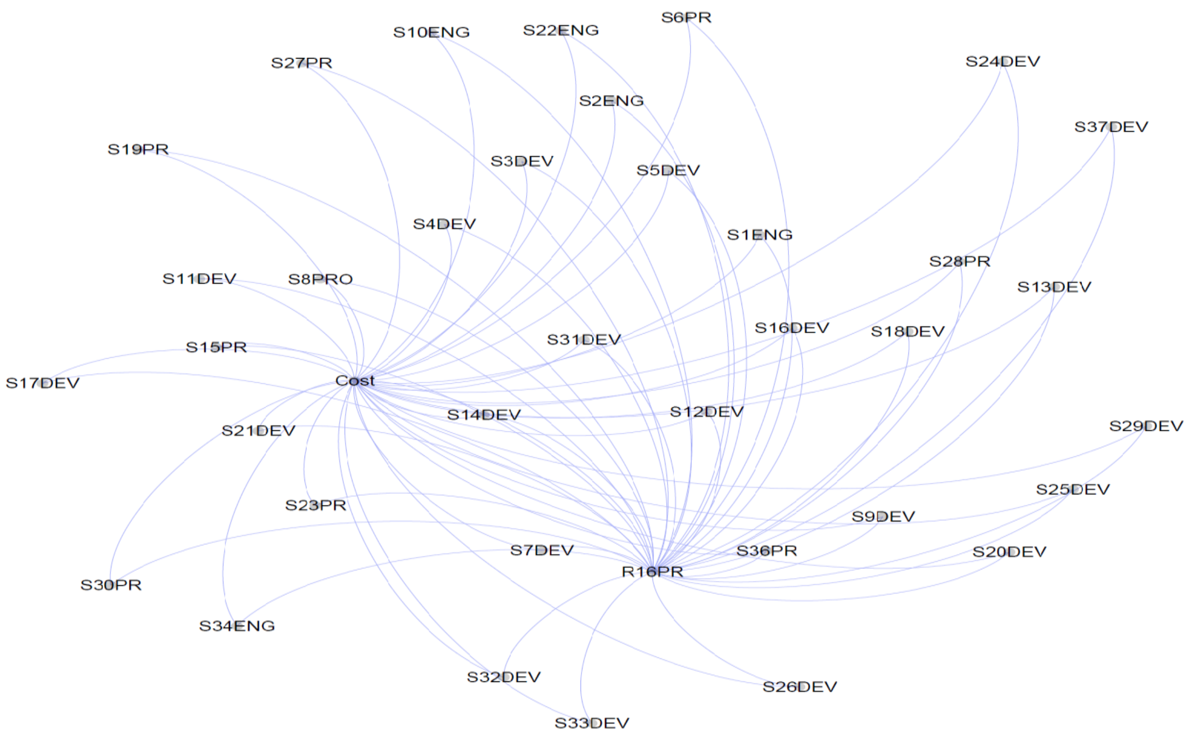


Figure 10-11 R16PR's influence on success factors in the cost ego network

As mentioned above, the betweenness centrality will be used to determine which risk factor has the most control over the success factors in the cost ego network. This research has found that R16PR Unrealistic budget is the most controlling factor with betweenness value of 35.5. Furthermore, this factor has interactions with 39.6% of the factors in this network. In addition, R16PR has interactions with 36 success factors. This research has also found that this factor is in close proximity to all the other success factors as it has a closeness value of 1.6, which in turn emphasises the importance of this factor as well as its impact on the other success factors.

10.4 Modelling of the relationship between the isolated network nodes

To measure the strength of the interaction or interdependency between independent variables software risk and success events in the cost ego network, the following multiple regression models and results are shown below.

Models

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to cost	.999[a]	.999	.998	.02185	.999	715.483	59	46	.000
Success to cost	.999[a]	.999	.998	.02238	.999	1645.238	37	69	.000

10.4.1 *Risk factors association model results*

In this research, multiple regression methods are used to predict the value of the dependent variable cost based on the value of risk events impact (independent variables). The results are used to explain the contribution of the factors to the cost of the project.

The summary for the sub-models shows the outputs of R Square, which measures the goodness of fit for the estimated regression equation for the sub-models. This measure explains or captures the amount of variation in the data captured by each equation. The Risk to cost sub-model is associated with a large number of risk impacts. There is a 99% association between Risk to cost and their risk impact events. Furthermore, approximately 1% of the risk events association cannot be explained by the model; this could be due to other Risk to cost factors that could not be added. The p-values shown above determine if the model is a good fit for the observed data. The p-values that are ≤ 5% mean that the null hypothesis can be rejected. As can be seen, the p-values in Risk to cost are 000% ≤ 5%, which rejects the null hypothesis. This model is likely to show a real representation of the association between dependent and independent variables.

Table 10-2 Risk factors association model results for Risk-to-cost

Model	Unstandardised Coefficients		Sig.	Model	Unstandardised Coefficients		Sig.
	B	Std. Error			B	Std. Error	
(Constant)	.031	.031	.325	R31ENG	.019	.005	.000
R1ENG	.022	.006	.001	R32DEV	.020	.006	.001
R2ENG	.001	.005	.859	R33ENG	.015	.005	.006
R3ENG	.031	.005	.000	R34ENG	.024	.006	.000
R4ENG	.011	.006	.064	R36ENG	.018	.006	.006
R5ENG	.017	.005	.001	R37DEV	.026	.005	.000
R6ENG	.012	.005	.026	R38DEV	.011	.007	.092
R7DEV	.016	.005	.005	R39PR	.020	.005	.001
R8DEV	.009	.005	.094	R40DEV	.017	.005	.001
R9DEV	.002	.006	.748	R42DEV	.018	.006	.003
R10DEV	.019	.005	.000	R43PR	.007	.006	.258
R11DEV	.021	.006	.000	R44ENG	.027	.006	.000

R12DEV	.016	.006	.005	R45ENG	.028	.006	.000
R13DEV	.011	.005	.032	R46ENG	.016	.006	.012
R14DEV	.008	.005	.112	R47DEV	-.001	.006	.812
R15PR	.027	.006	.000	R48ENG	.031	.006	.000
R16PR	.023	.006	.000	R49ENG	.024	.006	.000
R17PR	.022	.005	.000	R51PR	.012	.006	.031
R18PR	.017	.004	.000	R52ENG	.011	.006	.099
R19PR	.014	.004	.001	R53ENG	.017	.007	.017
R20ENG	.018	.005	.001	R54DEV	.008	.005	.109
R21ENG	.007	.006	.243	R55DEV	.013	.006	.042
R22DEV	.005	.005	.355	R56DEV	.031	.005	.000
R23DEV	.017	.005	.003	R57ENG	.021	.005	.000
R24PR	.021	.005	.000	R58ENG	.015	.005	.007
R26DEV	.020	.004	.000	R59PR	.014	.005	.004
R27DEV	.016	.005	.001	R61ENG	.028	.005	.000
R28ENG	.017	.005	.004	R62ENG	.025	.005	.000
R29ENG	.008	.006	.142	R64DEV	.025	.005	.000
R30PR	.024	.004	.000	S4DEV	-.001	.005	.816
	a. Dependent Variable: Risk-to-cost						

It noticeable that R3ENG Lack of technical skills, R48ENG Inconsistent coding style and R56DEV No backward compatibility and version management plan have the largest effect on project cost where, for every unit increase in those factors, a 0.031 unit increase in cost is predicted. Although, as has been mentioned, most software development project failure is not due to lack of technical skills, the reason behind the high score could be because managers give unnecessary focus to the technical skills as an critical and important factor to project success where, according to Jalil (2008), most project managers focus on the project outcome and technical skills, which results in cost overrun. The risk of R48ENG Inconsistent coding style to a project is due to its effect on the overall understanding of the software code, which in turn could affect the maintenance of the software as, according to Lee and Rhew (2014), “Software understanding is so important that it accounts for 50% of full efforts taken during software maintenance”. Using an inconsistent coding style is one of the most common risk factors in maintaining a software project (Thummalapenta et al., 2010).

It is also noticeable that two factors have a negative impact on cost in this model: first, R47DEV Design is skipped or created after code is written. The reason for its negative impact could be because the cost of the design is reduced temporarily until the code it written. This research found that this effect is quite small on the project cost, which is supported by its B value of .001. S4DEV Efficient project management is the second value that has a negative impact on software project cost, although this factor has a huge influence on the project cost in terms of its interaction. Its negativity could be mainly because of the question’s design as it looked at the influences of the success factors and success in relation to the success criteria where the rate of impact was assessed in two separate parts, one for the success and the other for the risk, as has been mentioned.

10.4.2 *Success factors association model results*

Table 10-3 Success factors association model results for success-to-cost

Model	Unstandardised Coefficients		Sig.	Model	Unstandardised Coefficients		Sig.
	B	Std. Error			B	Std. Error	
(Constant)	.013	.021	.536	S19PR	.019	.006	.003
S1ENG	.023	.005	.000	S20DEV	.028	.005	.000
S2ENG	.017	.005	.001	S21DEV	.033	.005	.000
S3DEV	.017	.005	.002	S22ENG	.031	.005	.000
S4DEV	.028	.005	.000	S23PR	.022	.004	.000
S5DEV	.022	.005	.000	S24DEV	.028	.005	.000
S6PR	.028	.005	.000	S25DEV	.024	.005	.000
S7DEV	.030	.005	.000	S26DEV	.031	.004	.000
S8PR	.042	.005	.000	S27PR	.025	.005	.000
S9DEV	.035	.006	.000	S28PR	.035	.005	.000
S10ENG	.018	.006	.002	S29DEV	.040	.005	.000
S11DEV	.028	.007	.000	S30PR	.028	.004	.000
S12DEV	.031	.005	.000	S31DEV	.033	.005	.000
S13DEV	.024	.005	.000	S32DEV	.038	.005	.000
S14DEV	.024	.004	.000	S33DEV	.036	.005	.000
S15PR	.024	.006	.000	S34ENG	.034	.005	.000
S16DEV	.016	.006	.005	S36PR	.034	.005	.000
S17DEV	.024	.004	.000	S37DEV	.030	.005	.000
S18DEV	.018	.005	.001	R16PR	.002	.004	.723
	a. Dependent Variable: Success-to-cost						

The results in confirm the rejection of the null (default) hypothesis that success factors have absolutely no effect on project cost. It seems that S8PR Realistic budget and S29DEV Team capabilities have the most effect on project cost. This shows the importance of having a realistic budget on the project cost as an increase of a unit of realistic budget has a positive impact on project cost with an increased value of 0.042. A software project without a realistic budget could find that its funds run out early in the project stage, which could lead to project failure or overspend where the same amount of money could be invested to build a successful project but with a realistic budget (Hijazi et al., 2014). It is interesting to note that S3DEV Realistic schedule has a relatively weak effect on project cost although it has been recognised as important to cost saving. The reason behind that could be because the project schedule in Saudi Arabia are not considered to have a huge effect on project cost due to its changing nature, and the results have recognised that S33DEV Project criticality has more effect on the cost.

10.5 The association between the network nodes

Table 10-4 S4DEV and Risk-to-cost association network

	S4DEV			Risk-to-cost		
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
R1ENG	.258**	.007		.348**	.000	R1ENG
R2ENG	.093	.343		.420**	.000	R2ENG
R3ENG	.195*	.044		.346**	.000	R3ENG
R4ENG	.093	.340		.410**	.000	R4ENG
R5ENG	.138	.157		.394**	.000	R5ENG
R6ENG	.229*	.018		.275**	.004	R6ENG
R7DEV	.044	.650		.475**	.000	R7DEV
R8DEV	-.028	.779		.439**	.000	R8DEV

R9DEV	.078	.422		.597**	.000	R9DEV
R10DEV	.172	.077		.378**	.000	R10DEV
R11DEV	.194*	.045		.367**	.000	R11DEV
R12DEV	.285**	.003		.555**	.000	R12DEV
R13DEV	.310**	.001		.591**	.000	R13DEV
R14DEV	.231*	.017		.472**	.000	R14DEV
R15PR	.121	.213		.454**	.000	R15PR
R16PR	.133	.173		.470**	.000	R16PR
R17PR	.127	.194		.602**	.000	R17PR
R18PR	.192*	.048		.478**	.000	R18PR
R19PR	.018	.854		.333**	.000	R19PR
R20ENG	.207*	.033		.599**	.000	R20ENG
R21ENG	.176	.069		.592**	.000	R21ENG
R22DEV	.345**	.000		.446**	.000	R22DEV
R23DEV	.323**	.001		.485**	.000	R23DEV
R24PR	.174	.073		.552**	.000	R24PR
R26DEV	.093	.340		.530**	.000	R26DEV
R27DEV	.265**	.006		.509**	.000	R27DEV
R28ENG	.213*	.028		.574**	.000	R28ENG
R29ENG	.191*	.049		.490**	.000	R29ENG
R30PR	.319**	.001		.575**	.000	R30PR
R31ENG	.363**	.000		.572**	.000	R31ENG
R32DEV	.235*	.015		.408**	.000	R32DEV
R33ENG	.152	.117		.381**	.000	R33ENG

R34ENG	.365**	.000		.682**	.000	R34ENG
R36ENG	.248*	.010		.661**	.000	R36ENG
R37DEV	.263**	.006		.509**	.000	R37DEV
R38DEV	.280**	.003		.584**	.000	R38DEV
R39PR	.222*	.021		.254**	.008	R39PR
R40DEV	.263**	.006		.539**	.000	R40DEV
R42DEV	.241*	.012		.508**	.000	R42DEV
R43PR	.262**	.006		.570**	.000	R43PR
R44ENG	.244*	.011		.446**	.000	R44ENG
R45ENG	.188	.053		.557**	.000	R45ENG
R46ENG	.070	.471		.631**	.000	R46ENG
R47DEV	.289**	.003		.480**	.000	R47DEV
R48ENG	.132	.176		.684**	.000	R48ENG
R49ENG	.290**	.002		.640**	.000	R49ENG
R51PR	.347**	.000		.624**	.000	R51PR
R52ENG	.202*	.037		.730**	.000	R52ENG
R53ENG	.254**	.008		.600**	.000	R53ENG
R54DEV	.197*	.042		.650**	.000	R54DEV
R55DEV	.121	.214		.674**	.000	R55DEV
R56DEV	-.014	.890		.558**	.000	R56DEV
R57ENG	.224*	.020		.567**	.000	R57ENG
R58ENG	.158	.104		.597**	.000	R58ENG
R59PR	.152	.118		.502**	.000	R59PR
R61ENG	.181	.063		.649**	.000	R61ENG

R62ENG	.216*	.025		.501**	.000	R62ENG
R64DEV	.218*	.024		.621**	.000	R64DEV
S4DEV	1			.375**	.000	S4DEV
Risk-to-cost	.375**	.000		1		Risk-to-cost

10.5.1 *Risk influence on cost*

This section looks at the correlation between the interacted factors in the Cost & S4DEV ego network to find the significant relationships to determine the strongest and the weakest relationship between them From Table 10-5 we can see the relationship between factors in risk to cost network factors was investigated using Pearson product-moment correlation coefficient. Preliminary analyses were performed to ensure no violation of the assumptions of normality, linearity and homoscedasticity. The results show that 35 factors are significantly correlated with S4DEV. Eighteen factors have a correlation that is significant at the 0.01 level and 28 factors have a correlation that is significant at the 0.05 level. There was a strong, positive correlation between the risk cost and S4DEV [r= 0.375, p<.0005]; this also supports the results that claim that S4DEV is the most central node in the network. R34ENG has the strongest and most positive relationship with S4DEV of all the risk factors. These results emphasise the importance of effective management on an incompatible development environment. Furthermore, S34ENG has a strong and positive correlation with Risk-to-cost [r= 0.682, p<.0005]. It is interesting to find that factors S31ENG Improper marketing techniques and S51PR Data Privacy Issues have the third and fourth strongest relations with S4DEV; this highlights the importance of the product marketing on the project cost. It is extremely interesting to see data privacy issues having a strong correlation with S4DEV; this is could be because data breaches are constantly on the rise, and so managers have to take into account issues that relate to privacy like compliance, security and approaches to protecting private data.

The results also show that there is a strong, positive correlation between the risk cost and R52ENG Insufficient consideration of reliability and availability [r= 0.730, p<.0005], with high levels of insufficient consideration of reliability and availability correlating with high levels of project cost. R48ENG Inconsistent coding style also has a positive and strong relationship with project cost [r= 0.684, p<.0005] as inconsistent coding styles cause programmers to work more slowly when writing codes, which in turn will result in extra time and cost to a project (Goodliffe, 2014). It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised as having a correlation that is significant at the 0.01 level with the risk-to-cost, which agrees with this research's claim of their importance to what have been mentioned as important to the cost ego network as well as their interaction with S4DEV.

10.5.2 *Success influencing cost*

Table 10-5 R16PR and success-to-cost association network

Factors	Success-to-cost			R16PR		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
S1ENG	.570**	.000		.240*	.013	S1ENG
S2ENG	.683**	.000		.191*	.049	S2ENG
S3DEV	.656**	.000		.155	.110	S3DEV
S4DEV	.664**	.000		.133	.173	S4DEV
S5DEV	.638**	.000		.252**	.009	S5DEV
S6PR	.696**	.000		.292**	.002	S6PR
S7DEV	.658**	.000		.363**	.000	S7DEV

S8PR	.610**	.000		.463**	.000	S8PR
S9DEV	.669**	.000		.128	.189	S9DEV
S10ENG	.722**	.000		.028	.773	S10ENG
S11DEV	.792**	.000		.176	.070	S11DEV
S12DEV	.738**	.000		.082	.402	S12DEV
S13DEV	.696**	.000		.142	.145	S13DEV
S14DEV	.665**	.000		.361**	.000	S14DEV
S15PR	.607**	.000		.236*	.014	S15PR
S16DEV	.699**	.000		.153	.116	S16DEV
S17DEV	.603**	.000		.123	.208	S17DEV
S18DEV	.719**	.000		.076	.436	S18DEV
S19PR	.662**	.000		.134	.168	S19PR
S20DEV	.680**	.000		.148	.127	S20DEV
S21DEV	.702**	.000		.104	.285	S21DEV
S22ENG	.644**	.000		.124	.204	S22ENG
S23PR	.645**	.000		.137	.159	S23PR
S24DEV	.765**	.000		.056	.564	S24DEV
S25DEV	.618**	.000		.115	.240	S25DEV
S26DEV	.654**	.000		.128	.191	S26DEV
S27PR	.604**	.000		-.086	.377	S27PR
S28PR	.703**	.000		.051	.600	S28PR

S29DEV	.724**	.000		-.022	.822	S29DEV
S30PR	.540**	.000		.231*	.017	S30PR
S31DEV	.721**	.000		.059	.545	S31DEV
S32DEV	.766**	.000		.237*	.014	S32DEV
S33DEV	.606**	.000		.035	.717	S33DEV
S34ENG	.539**	.000		-.081	.405	S34ENG
S36PR	.718**	.000		.076	.435	S36PR
S37DEV	.659**	.000		.093	.343	S37DEV
Success-to-cost	1			.217*	.025	Success-to-cost
R16PR	.217*	.025		1		R16PR

This section looks at the correlation between the interacting factors in the success-to-cost network to find the significant relationships to determine the direction of the relationship, the strength of the relationship and the weak relationship between them.

From we can see the relationship between factors in the success-to-cost network factors was investigated using Pearson product-moment correlation coefficient. Preliminary analyses were performed to ensure no violation of the assumptions of normality, linearity and homoscedasticity. The results show that 11 factors are significantly correlated with R16PR. There are five factors that have a correlation that is significant at the 0.01 level and six factors with a correlation that is significant at the 0.05 level. There was a strong, positive correlation between R16PR Unrealistic budget and S8PR Realistic budget [r= 0. 463, p<.0005]. These results are considered to be important as having a realistic budget will definitely affect the project’s budget. However, this research argues that this correlation is supposed to be a negative relationship as an increase in one factor should result in a decrease in the other factor.

The reason why the correlation is not in the right direction could be the design of the question, as the aim of the question was to find the effect of risk and success factors on the success criteria without looking directly at their effects on each other. Also, it is noted that S8PR has a stronger and more positive relationship with Success-to-cost [r= 0. 610, p<.0005] than R16PR [r= 0. 217, p<.0005], which makes it the weakest significant correlation with Success-to-cost . Furthermore, S7DEV has the second strongest and positive relationship with S16PR [r= 0. 363, p<.0005]. It is interesting to find that the factor Success-to-cost has one of the weakest relationships with R16PR [r= 0. 217, p<.0005] although it is the most central node in the network. This result shows that even R16PR is not considered to form the strongest relationship with Success-to-cost but its indirect correlation with Success-to-cost is what makes it important to Success-to-cost . This research also notes that R16PR has three factors that have a negative relationship: S27PR Effective contract management, S34ENG Getting code from quality reliable and stable community and S29DEV Team capability. However, because of their insignificant values, they have not been taken into account as having a strong or a weak relationship.

The results also show that there was a strong, positive correlation between Success-to-cost and S11DEV Appropriate development process/methodology” [r= 0. 792, p<.0005]. This result agrees with what Verma et al. (2014) have noted: that in software development different methodologies will have different effects on the project cost. It is worth mentioning that all the correlations in this network have a positive relationship where they have all been recognised as having a correlation that is significant at the 0.01 level with the Success-to-cost , which meets this research’s claim of their importance to what have been mentioned as important to the cost ego network as well as their interaction with R16PR. The only factor that has a correlation that is significant at the 0.05 level is R16PR. As mentioned, this could be because the network was about the influence of the most central risk factor where all the factors have been correlated from a success point of view to success-to-cost, which results inR16PR having a huge but indirect relationship with success-to-cost.

10.6 **Summary**

In this chapter, the cost network was the ego topology used to study the impact of the success and risk factors on it. The top central factors were identified and analysed. This chapter has investigated the contribution of the factors in cost prediction, and analysed the association between these factors.

Chapter 11: Quality ego network

Chapter 11: Quality ego network

11.1 Introduction

In this chapter, the research explores the interdependency of the success and risk factors in the quality ego network. This chapter starts by describing the ego network general characteristics and top central factors from the success and risk factors. Then the top central factors are isolated to study their contribution to the quality ego network; the associations between the factors are also explored and analysed.

11.2 Quality ego network

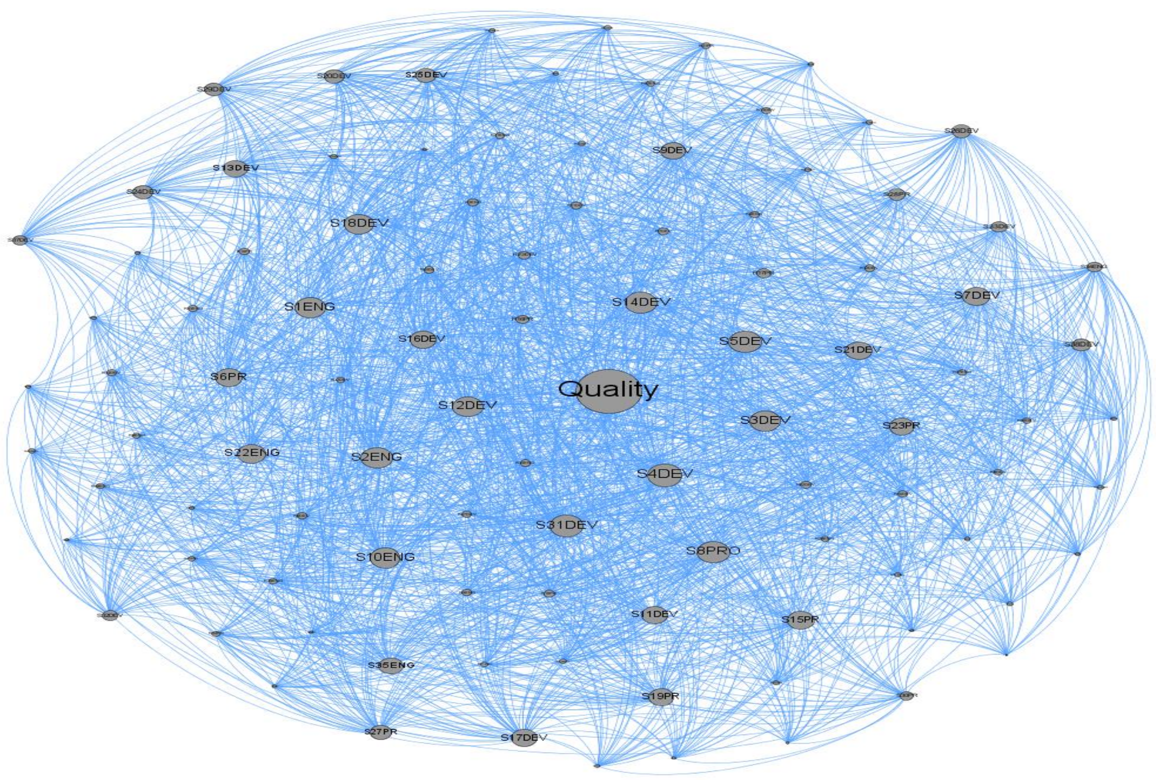


Figure 11-1 Quality ego network

In this network, there are 102 factors connected directly to Quality out of 106 possible connections, which shows that quality is the most connecting criteria among the four criteria. Only four nodes are not directly connected to Quality. The other three criteria have been isolated and removed from the network to study the interaction of the risk and success factors in relation to Quality without the influence of the other success criteria, in order to obtain the maximum interaction impact for Quality itself.

The centrality measures are the focus of the Quality ego network degree as they show the number of interactions in the network. Closeness shows how close a factor is to every other factor in the network. Betweenness will be used to determine the centrality of the factors if they have the same degree and closeness centrality.

11.2.1 *Top five central factors influencing quality*

Table 11-1 Top five central success factors in quality ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
S4DEV	65	1.356436	160.8538	0.594542	1
S31DEV	63	1.376238	149.7847	0.578484	2
S8PR	62	1.386139	144.9712	0.5696	3
S5DEV	62	1.386139	143.5473	0.57158	4
S2ENG	61	1.39604	140.3492	0.56008	5
S14DEV	61	1.39604	138.83	0.562732	5

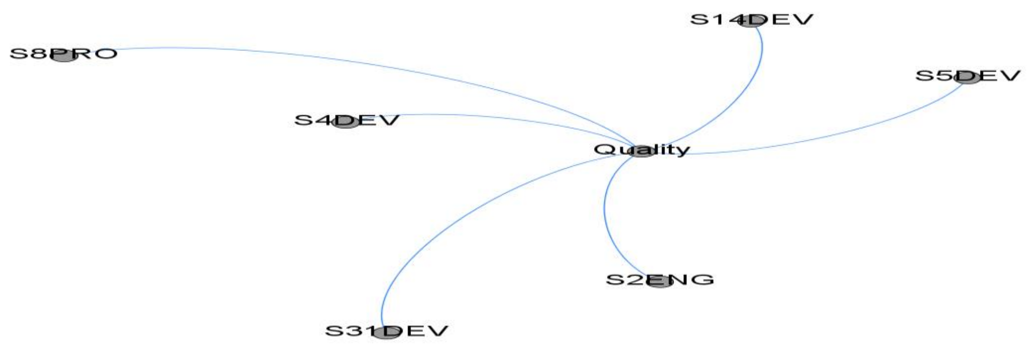


Figure 11-2 Top five central factors influencing quality

S4DEV Efficient project management is the most central factor in terms of degree to the Quality ego network with a degree of 65 connections. It is directly connected to the largest number of factors. Therefore, this factor is considered to be a central factor in terms of its direct impact on other Quality factors or, in other words, its impact can spread directly to the majority of network risk and success factors. This means that anything that affects efficient project management will have a huge impact on the Quality. For example, higher software quality depends on the software project having a project manager who has the skills and desire to produce a high-quality product as well “lower costs and higher client satisfaction” (Langer et al., 2008). Also, Varajão et al. (2014) noted that efficient project management is connected to many software development project aspects, for example, "top managing involvement, proper project planning, well-defined requirements, frequent checkpoint controls, skilled working teams, and team project effort". These are very important factors in determining the software project quality and are important to the way in which efficient project management affects the overall project success. Recently, many software companies in the USA and UK have outsourced their software projects to certain Asian countries where, according to Akbar et al. (2012), efficient project management is considered as a silver lining between the success and failure of a project.

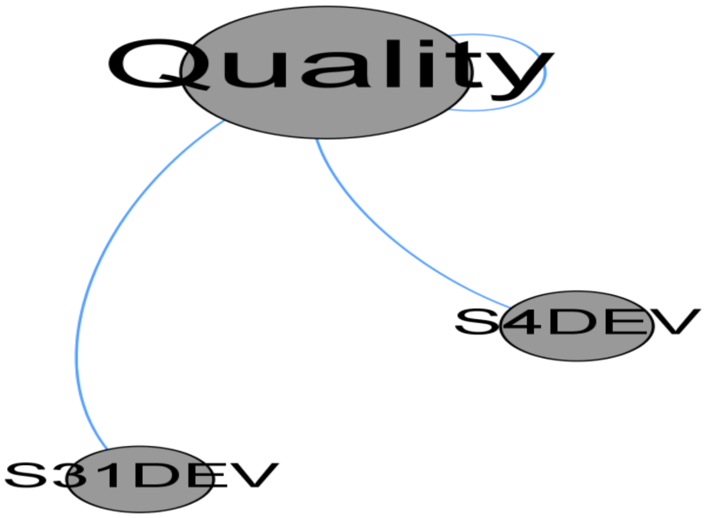


Figure 11-3 Quality ego network with a degree of 63

S31DEV Organisational culture is the second factor in terms of its impact on Quality, as it has 63 connections. According to Dahlgaard et al. (2006), it is important to gain customer satisfaction and improve the project quality by giving more focus to organisational culture. Kim, Pindur and Reynolds (1995) and Hildbrandt et al. (1991) have also noted that it is essential to recognise organisational cultural as a primary condition for the successful implementation of quality management in a project. According Budhwar and Mathew (2007)"Organisational culture would have a positive relationship with quality in a knowledge intensive work environment such as software"; they found that organisational culture has played an important role in making quality requirements an important part of organisational life.

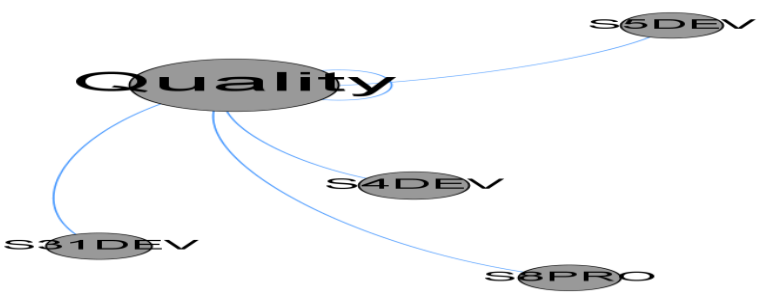


Figure 11-4 Quality ego network with a degree of 62

S8PR Realistic budget and S5DEV Top level management support have the degree centrality of 62 connections and closeness centrality value of 1. 39. This shows that they are both connected to 62 factors in the Quality ego network. However, S8PR Realistic budget is more central than S5DEV Top level management support because it has a higher number of shortest paths with betweenness value of 144.97; thus, it realistic budget is considered as the third factor in terms of its influence on Quality. Furthermore, Realistic budget has connections to many factors in software development where some researchers have gone to the extent of considering team distribution and team size as sub-factors to Realistic budget (Darwish and Rizk, 2015). On the other hand, S5DEV Top level management support has been ranked as the fourth most central node in the quality ego network. According to Varajão et al. (2014), top management involvement has been recognised as one of the top five success factors, comparing it to other types of projects, like construction projects, where it ranked as the 10th most important factor. Software projects are not always about software and hardware; for example, Shi (2010) noted that communication with top management is a decisive factor in software project success. The importance of this factor is based on its effect on organisational culture, where it is considered a more central node to quality than S8PR as, according to Ahimbisibwe et al. (2015), "Some of the factors that are affected by top-level management, leadership, strategic direction and client organisational culture". A traditional software methodology is more like to succeed if top level management support is implemented (Sheffield and Lemétayer, 2013).

The fifth central factor is S2ENG Clear objectives and goals as it has connections to 61 risk factors and has closeness value of 1.40. Also, it has the fifth highest betweenness value of 140.35, which emphasises the importance as it has a big effect on project quality. Furthermore, S14DEV has the same centrality as S2ENG in terms of its degree and closeness centrality but it has a lower betweenness value of 140.35.

11.2.2 *Top five risk influencing quality*

This research investigated the top central risk factors in this network, and the top five are shown in Table 11-2.

Table 11-2 Top five central Risk factors in Quality ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
R17PR	38	1.623762	32.32894	0.452399	1
R16PR	37	1.633663	30.27344	0.442556	2
R23DEV	36	1.643564	28.29259	0.432311	3
R11DEV	35	1.653465	26.97751	0.419461	4
R3ENG	35	1.653465	26.12742	0.423874	5

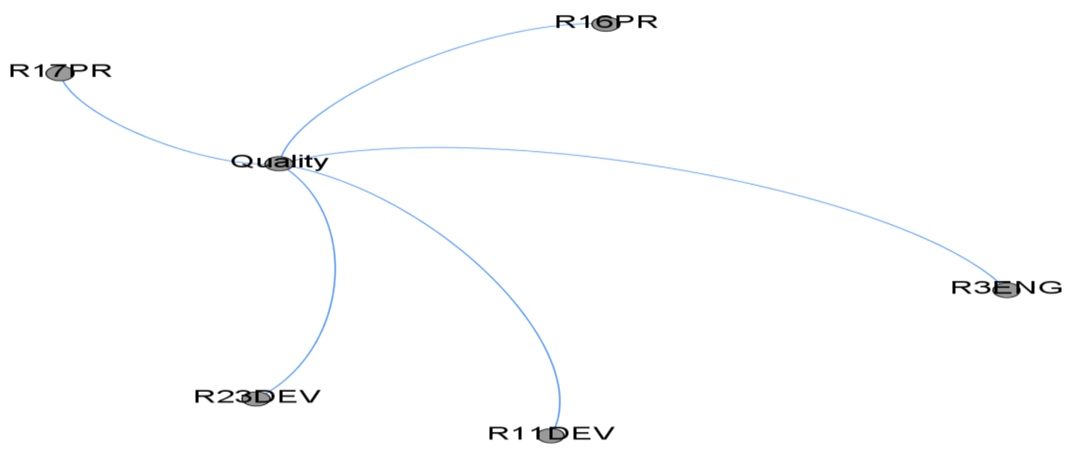


Figure 11-5 Top five risks influencing quality

R17PR Resource insufficiency is considered as the most central factor to the quality ego network as it has a direct impact on 38 success factors and the lowest average distance to all the other factors with a closeness value of 1.62. Likewise, it has the highest number of shortest paths to all other factors in the network with a betweenness value of 32.33, which indicates that it has more control in the quality ego network.

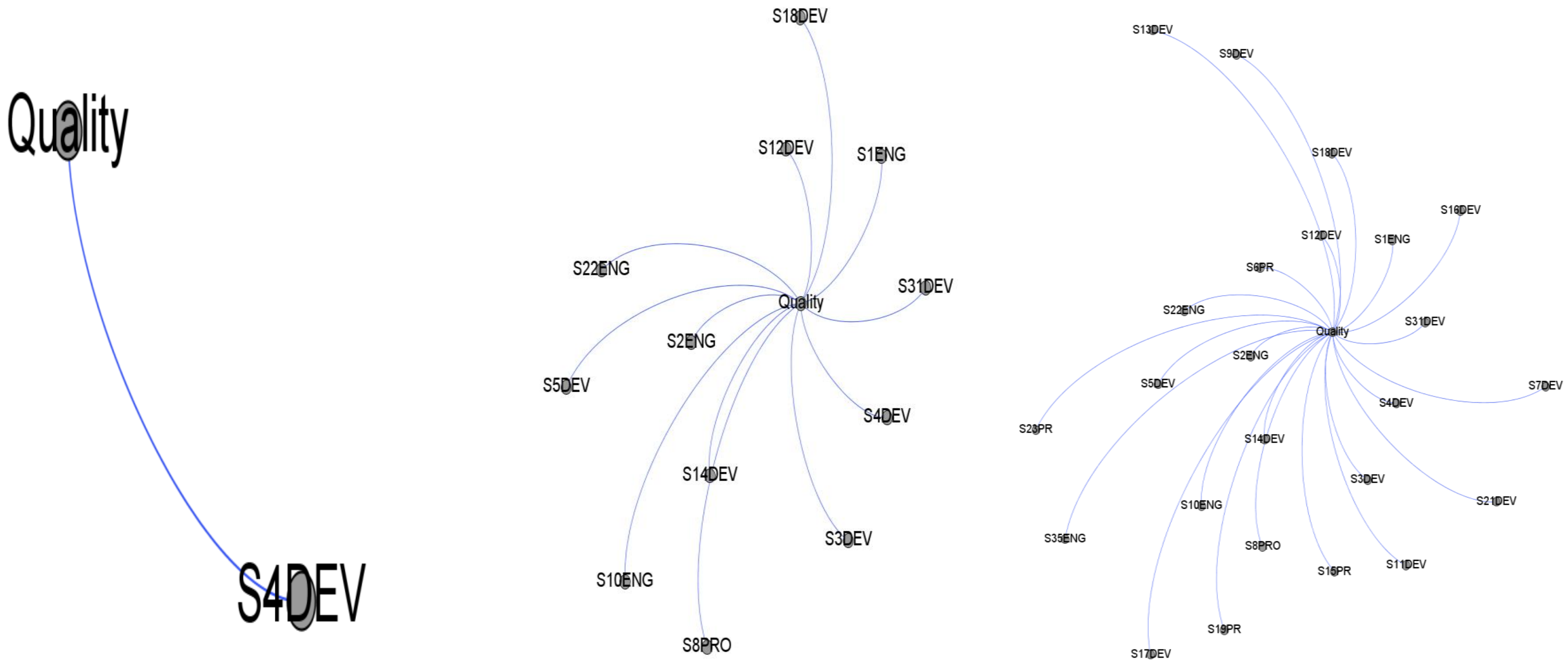
R16PR Unrealistic budget is considered as the second most central factor to the quality network as it has a direct impact on 37 success factors and the lowest average distance to all the factors with a closeness value of 1.63. Furthermore, it has a betweenness value of 30.27.

R23DEV L Project manager lacks experience is the third most central factor in the quality ego network with a degree of 36 connections and closeness centrality value of 1.64.

R11DEV Unrealistic resource planning and R3ENG Lack of technical skills are the fourth and fifth centrality risk factors. Although they have the same number of connections to other factors with a degree of 35 and the same distance to all other factors with a closeness of 1.65, R11DEV has both more shortest paths and more control of the factors' interactions with each other with a betweenness value of 26.98 compared to R3ENG with a betweenness centrality value of 26.13.

11.3 Isolating the risk and success factors related to the Quality ego network

The aim of this section is to find the factors controlling the influence of risk factors over the success factors. The same three-step methodology used in the cost ego network is applied to identify the most controlling factors from the success and risk factors. After that the contraption and production of those factors to quality, will be explored. The associations between the factors are reported as well.

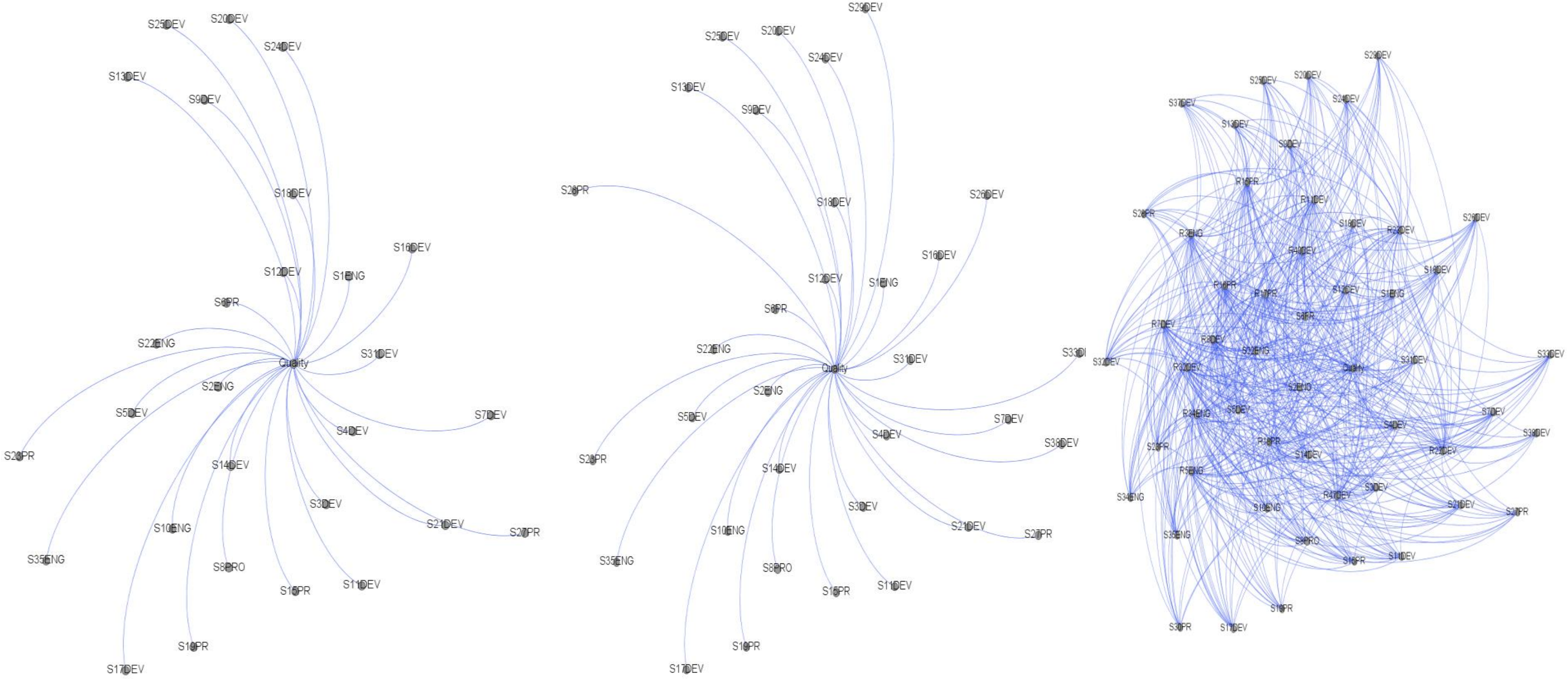


This graph shows the success factors with a degree of 64 and more in the quality ego network

This graph shows the success factors with a degree of 57 and more in the quality ego network

This graph shows the success factors with a degree of 52 and more in the quality ego network

Figure 11-6 The change that occurs in the quality ego network and its success factors when adjusting the degree topology value part one



This graph shows the success factors with a degree of 47 and more in the quality ego network

This graph shows the success factors with a degree of 42 and more in the quality ego network

This graph shows the success factors with a degree of 34 and more in the quality ego network

Figure 11-7 The change that occurs in the quality ego network and its success factors when adjusting the degree topology value part two

11.3.1 *Risk factors influencing QUALITY*

The algorithm has been applied in various steps. In the degree range of 64 it is noticeable that the most central factor is S4DEV Effective project management. When reducing the minimum degree range to 57, the number of factors connected to QUALITY increased to 12. In the minimum degree range of 52, there are 25 factors connected to the QUALITY ego network. Twenty-eight factors are respectively central to QUALITY, which is an increase of 16%, when the degree of 47 is applied. As can be seen above, there are 33 factors with a degree of 42 or higher. The last degree range applied in the QUALITY ego network is 34, with 52 factors interacting with QUALITY directly.

11.3.1.1 *The success factor that has the most control and influence over the risk factors in quality ego network*

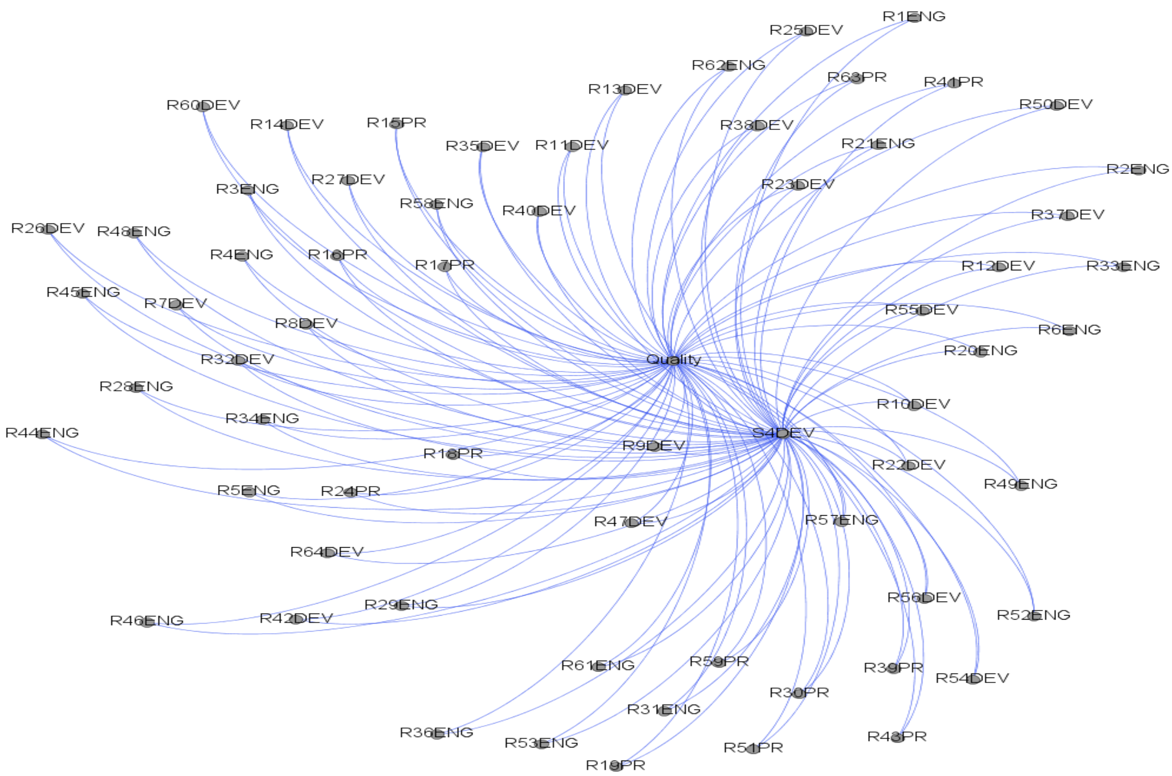
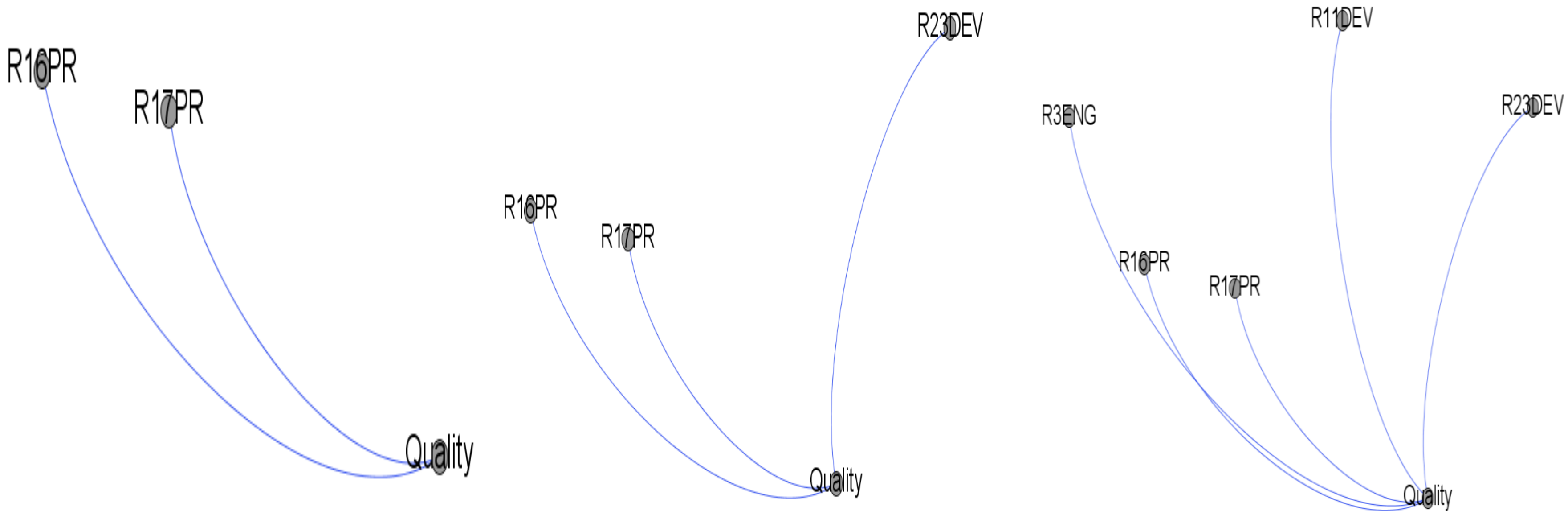


Figure 11-8 S4DEV: the success factor that has the most influence on risk factors in the quality ego network

It is noticeable that the most central node in terms of its closeness is S4DEV Effective project management; also, the number of factors in this degree range are 62.7% of the total number of nodes available in this network. S4DEV is also the most central factor in the network in terms of its betweenness value of 160.86, where it is important to mention that S4DEV has more influence on quality than cost. Furthermore, as S4DEV is connected to 64 risk factors connected to the QUALITY ego network, effective project management has control over and influences all of the risk factors connecting to the QUALITY ego network. It should also be noted that this factor is in close proximity to the rest of the risk factors. This is supported by the value of its Closeness centrality of 1.36, which reflects the spread time and speed of this factor's impact on the other factors.. This result shows the importance of project quality and project management: where is a project successful if it is delivered on time and within the planned budget but the end product is of little use to the client? Questions like that demonstrate the importance of effective project management to achieve product quality through effective intra-organisational integration and optimal utilisation of scarce resources (Cicmil, 1997).

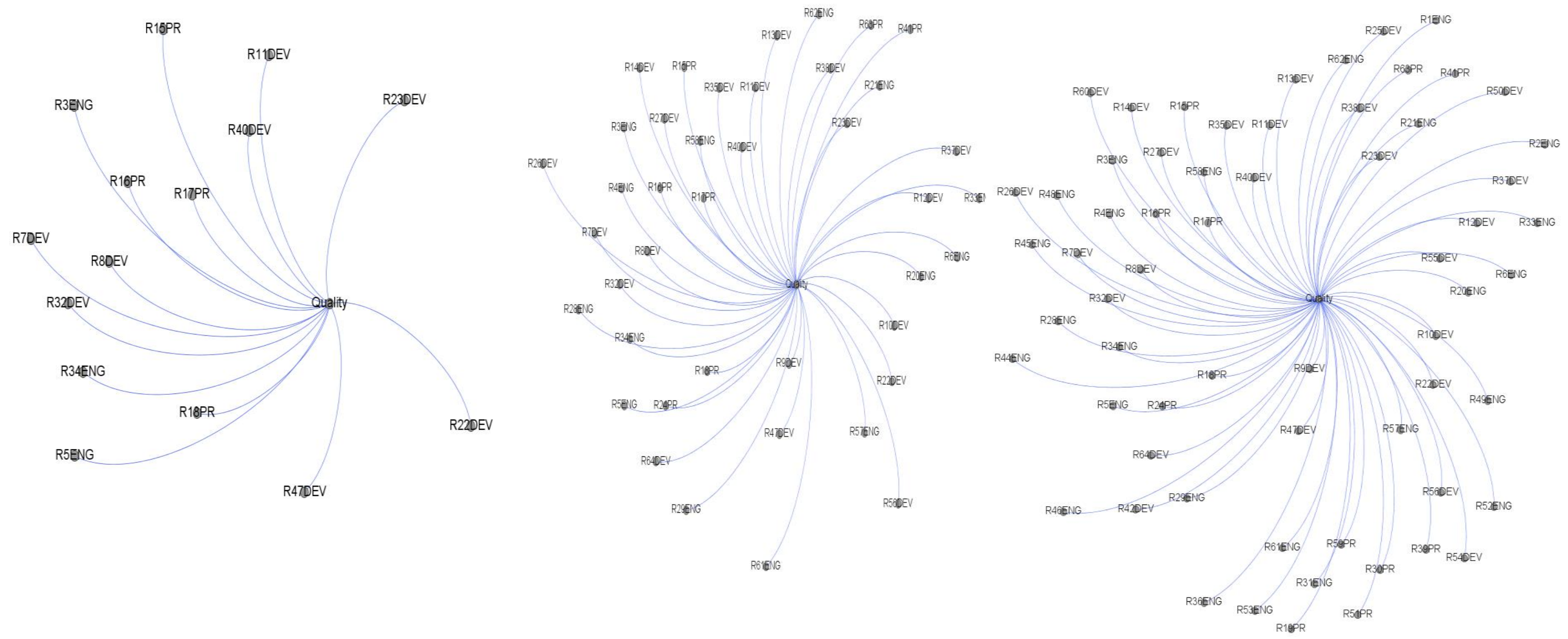


This graph shows the risk factors with a degree of 38 and more in the quality ego network

This graph shows the risk factors with a degree of 36 and more in the quality ego network

This graph shows the risk factors with a degree of 35 and more in the quality ego network

Figure 11-9 The change that occurs in the quality ego network and its risk factors when adjusting the degree topology value part one



This graph shows the risk factors with a degree of 34 and more in the quality ego network

This graph shows the risk factors with a degree of 30 and more in the quality ego network

This graph shows the risk factors with a degree of 25 and more in the quality ego network

Figure 11-10 The change that occurs in the quality ego network and its risk factors when adjusting the degree topology value part two

11.3.2 *Success factors influencing QUALITY*

The algorithm has been applied in various steps. In the degree range of 37, it can be seen from the graph above that there are two factors connected to QUALITY, R16PR Unrealistic budget and R17PR Resource insufficiency. When reducing the minimum degree range to 36, the number of success factors connected to QUALITY increased to three. In the minimum degree range of 35, there are five factors connected to the QUALITY ego network. Sixteen factors are respectively connected to QUALITY when the degree of 34 is applied. As can be seen above, that there are 42 factors with a degree of 30 or higher. The last degree range applied in the QUALITY ego network is the minimum degree range of 25 with 64 factors interacting with QUALITY directly.

11.3.2.1 *The risk factor that has the most control and influence over the success factors in quality ego network*

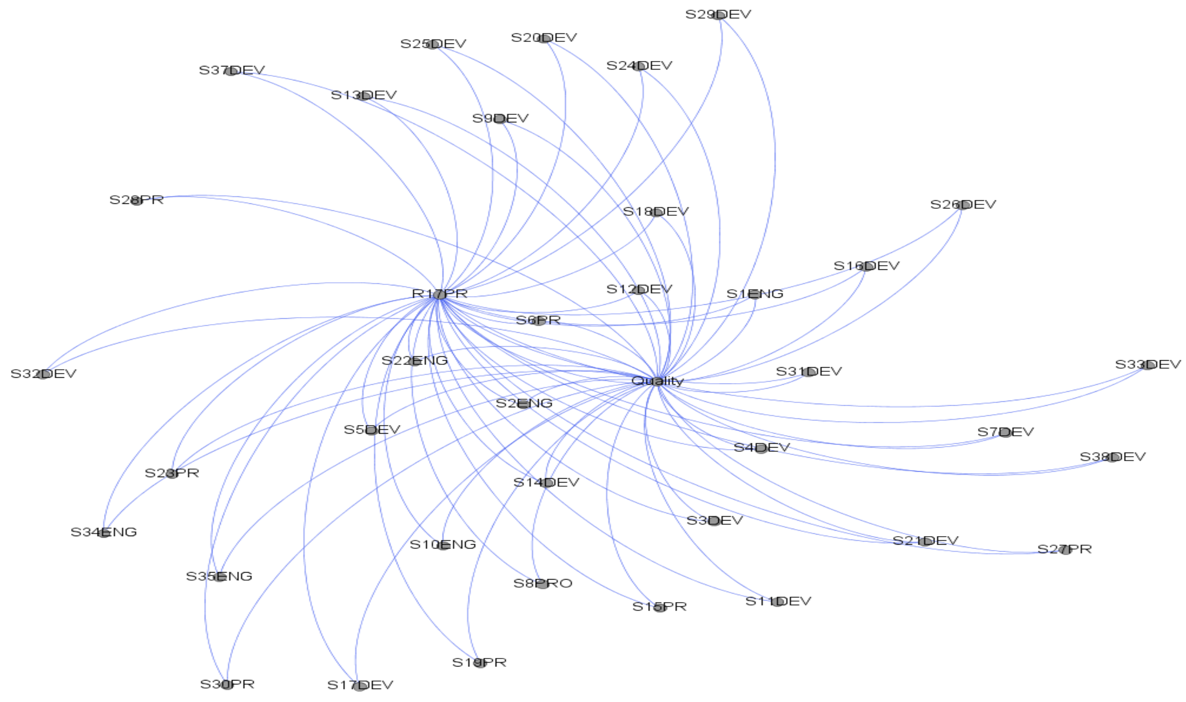


Figure 11-11R43PR's influences on success factors in the QUALITY ego network

It is noticeable that, in terms of betweenness, the most central node is R17PR Resource insufficiency . Furthermore, R17PR is connected to 37 success factors in the ego risk network but it is not connected to S36PR Commitment of stakeholders. R17PR Resource insufficiency is the most central factor in terms of closeness in the QUALITY ego network. It should also be noted that this factor is in close proximity to the rest of the success factors. This is supported by its closeness centrality value of 1.62, which reflects the spread time and speed of this factor's impact on the other factors.

11.4 Modelling of the relationship between the isolated network nodes

Models

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to QUALITY	1.000a	1.000	1.000	.00306	1.000	30916.590	65	41	.000
Success to QUALITY	1.000[a]	1.000	.999	.01389	1.000	3684.780	38	68	.000

11.4.1 *Risk factors association model results*

Multiple regression methods are used to predict the value of a proxy variable (dependent variable) based on the value of the impact of risk and success events (independent variables). This measure explains or captures the amount of variation in the data captured by each equation where the R square value in the sample tends to be a rather optimistic overestimation of the true value in the population. "The Adjusted R square statistic 'corrects' this value to provide a better estimate of the true population value" (Pallant, 2001).

The Risk to QUALITY sub-model is associated with a large number of risk impacts. There is a 100% association between Risk to QUALITY and its risk impact events, which reflects the influence of S4DEV on the risk factors' interaction with success. This is because all the risk factors have been included in this interaction network. The p-values shown above determine how good a fit the model is for the observed data. The p-values that are ≤ 5% mean that the null hypothesis can be rejected. As can be seen, the p-values in Risk to QUALITY are 000% ≤ 5%, which rejects the null hypothesis. This model is likely to show a real representation of the association between dependent and independent variables.

The success to quality sub-model is also associated with a large number of risk impacts. There is a 100% association between Risk to QUALITY and their risk impact events. However, it is worth mentioning that the Adjusted R square is 99%. The main reason why this value is different to the R square could relate to the absence of S36PR Commitment of stakeholders. As can be seen, the p-values in Risk to QUALITY are 000% ≤ 5%, which rejects the null hypothesis.

Table 11-3 Risk factors association model results for Risk-to-quality

Model	Unstandardised Coefficients			Model	Unstandardised Coefficients		
	B	Std. Error	Sig.		B	Std. Error	Sig.
(Constant)	.009	.004	.048	R33ENG	.0149	.001	.000
R1ENG	.0155	.001	.000	R34ENG	.0162	.001	.000
R2ENG	.0160	.001	.000	R35DEV	.0153	.001	.000
R3ENG	.0145	.001	.000	R36ENG	.0151	.001	.000
R4ENG	.0162	.001	.000	R37DEV	.0176	.001	.000
R5ENG	.0132	.001	.000	R38DEV	.0150	.001	.000
R6ENG	.0163	.001	.000	R39PR	.0158	.001	.000
R7DEV	.0148	.001	.000	R40DEV	.0161	.001	.000
R8DEV	.0161	.001	.000	R41PR	.0147	.001	.000
R9DEV	.0147	.001	.000	R42DEV	.0154	.001	.000

R10DEV	.0156	.001	.000	R43PR	.0149	.001	.000
R11DEV	.0163	.001	.000	R44ENG	.0146	.001	.000
R12DEV	.0161	.001	.000	R45ENG	.0153	.001	.000
R13DEV	.0159	.001	.000	R46ENG	.0168	.001	.000
R14DEV	.0168	.001	.000	R47DEV	.0152	.001	.000
R15PR	.0145	.001	.000	R48ENG	.0158	.001	.000
R16PR	.0146	.001	.000	R49ENG	.0175	.001	.000
R17PR	.0178	.001	.000	R50DEV	.0137	.001	.000
R18PR	.0165	.001	.000	R51PR	.0152	.001	.000
R19PR	.0150	.000	.000	R52ENG	.0156	.001	.000
R20ENG	.0157	.001	.000	R53ENG	.0154	.001	.000
R21ENG	.0143	.001	.000	R54DEV	.0136	.001	.000
R22DEV	.0174	.001	.000	R55DEV	.0176	.001	.000
R23DEV	.0159	.001	.000	R56DEV	.0150	.001	.000
R24PR	.0139	.001	.000	R57ENG	.0162	.001	.000
R25DEV	.0158	.000	.000	R58ENG	.0164	.001	.000
R26DEV	.0151	.001	.000	R59PR	.0158	.001	.000
R27DEV	.0154	.001	.000	R60DEV	.0162	.001	.000
R28ENG	.0169	.001	.000	R61ENG	.0149	.001	.000
R29ENG	.0148	.001	.000	R62ENG	.0157	.001	.000
R30PR	.0153	.001	.000	R63PR	.0162	.001	.000
R31ENG	.0155	.001	.000	R64DEV	.0162	.001	.000
R32DEV	.0156	.001	.000	S4DEV	-.0001	.001	.881
	a. Dependent Variable: RISK-TO-QUALITY						

The next thing we want to know is which of the variables included in the model contributed to the prediction of the dependent variable. In this case we are interested in comparing the contribution of each independent variable; therefore we will use the beta values.

It noticeable that R17PR Resource insufficiency has the largest beta coefficient of 0.178, which makes this factor the strongest unique contributor in explaining project QUALITY. For every unit increase in R17PR, a 0.178 unit increase in QUALITY is predicted. Also, the Beta value for Total R55DEV, R37DEV and R49ENG was slightly lower (0.176), indicating that it made less of a contribution. All the risk factors in this multiple regression have a positive impact on QUALITY. It is also noticeable that one factor has a negative impact on QUALITY in this model, S4DEV Effective project management, which may be because of the question's design as it looked at how success factors impact on project quality from a positive perspective, while it investigates the impact of the risk factors on quality from a negative perspective as it has been designed in two sections, as has been mentioned.

11.4.2 *Success factors association model results*

Table 11-4 Success factors association model results for success-to-quality

Model	Unstandardised Coefficients		Sig.	Model	Unstandardised Coefficients		Sig.
	B	Std. Error			B	Std. Error	
(Constant)	.0074	.013	.560	S20DEV	.0317	.003	.000
S1ENG	.0279	.003	.000	S21DEV	.0245	.003	.000
S2ENG	.0276	.004	.000	S22ENG	.0220	.003	.000
S3DEV	.0279	.003	.000	S23PR	.0250	.003	.000
S4DEV	.0270	.004	.000	S24DEV	.0374	.003	.000
S5DEV	.0297	.003	.000	S25DEV	.0276	.003	.000
S6PR	.0223	.003	.000	S26DEV	.0266	.002	.000

S7DEV	.0257	.003	.000	S27PR	.0235	.002	.000
S8PR	.0221	.003	.000	S28PR	.0255	.004	.000
S9DEV	.0205	.004	.000	S29DEV	.0345	.003	.000
S10ENG	.0240	.003	.000	S30PR	.0277	.002	.000
S11DEV	.0202	.003	.000	S31DEV	.0283	.003	.000
S12DEV	.0224	.003	.000	S32DEV	.0281	.003	.000
S13DEV	.0285	.003	.000	S33DEV	.0254	.003	.000
S14DEV	.0301	.003	.000	S34ENG	.0313	.003	.000
S15PR	.0244	.004	.000	S35ENG	.0298	.003	.000
S16DEV	.0234	.004	.000	S37DEV	.0301	.003	.000
S17DEV	.0224	.003	.000	S38DEV	.0329	.003	.000
S18DEV	.0341	.003	.000	R17PR	.0025	.003	.361
S19PR	.0242	.003	.000				
	a. Dependent Variable: SUCCESS-TO-QUALITY						

From the table above, it seems that S24DEV Clear assignment of roles and responsibilities makes the greatest contribution to project QUALITY with a value of 0.0374. This result shows the importance of clarity in people's understanding of their parts and type of involvement in a project and how this affects the overall project quality, as an increase in unit of S24DEV has a positive impact on project QUALITY with an increased value of 0.0374. It is interesting to note that S29DEV Team capabilities has relatively the second strongest contribution to quality as it has also been recognised to have a positive and strong contribution to project cost, which shows this factor's overall contribution to project success. It is noticeable that all the factors have a positive contribution to quality and it is interesting to find that all of them share the contribution to quality, with only small differences between them.

11.5 The association between the network nodes

Table 11-5 S4DEV and Risk-to-quality association network

Factors	RISK-TO-QUALITY			S4DEV		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
R1ENG	.292**	.002		.183	.059	R1ENG
R2ENG	.403**	.000		.145	.136	R2ENG
R3ENG	.317**	.001		.243*	.012	R3ENG
R4ENG	.489**	.000		.230*	.017	R4ENG
R5ENG	.399**	.000		.227*	.019	R5ENG
R6ENG	.370**	.000		-.001	.989	R6ENG
R7DEV	.479**	.000		.356**	.000	R7DEV
R8DEV	.383**	.000		.052	.593	R8DEV
R9DEV	.473**	.000		.208*	.032	R9DEV
R10DEV	.399**	.000		.134	.168	R10DEV
R11DEV	.472**	.000		.038	.698	R11DEV
R12DEV	.481**	.000		.140	.151	R12DEV
R13DEV	.535**	.000		.287**	.003	R13DEV
R14DEV	.350**	.000		.448**	.000	R14DEV
R15PR	.390**	.000		.280**	.004	R15PR
R16PR	.457**	.000		.290**	.002	R16PR

R17PR	.594**	.000		.382**	.000	R17PR
R18PR	.555**	.000		.246*	.011	R18PR
R19PR	.256**	.008		.074	.451	R19PR
R20ENG	.421**	.000		.146	.134	R20ENG
R21ENG	.654**	.000		.212*	.028	R21ENG
R22DEV	.519**	.000		.226*	.019	R22DEV
R23DEV	.413**	.000		.373**	.000	R23DEV
R24PR	.601**	.000		.095	.328	R24PR
R25DEV	.335**	.000		.036	.710	R25DEV
R26DEV	.326**	.001		-.064	.512	R26DEV
R27DEV	.583**	.000		.246*	.010	R27DEV
R28ENG	.515**	.000		.039	.693	R28ENG
R29ENG	.548**	.000		.143	.142	R29ENG
R30PR	.576**	.000		.026	.791	R30PR
R31ENG	.564**	.000		.140	.149	R31ENG
R32DEV	.471**	.000		.152	.118	R32DEV
R33ENG	.570**	.000		.249**	.010	R33ENG
R34ENG	.697**	.000		.232*	.016	R34ENG
R35DEV	.548**	.000		.104	.286	R35DEV
R36ENG	.458**	.000		.101	.301	R36ENG
R37DEV	.446**	.000		-.039	.693	R37DEV

R38DEV	.507**	.000		.053	.584	R38DEV
R39PR	.464**	.000		.117	.229	R39PR
R40DEV	.526**	.000		.245*	.011	R40DEV
R41PR	.368**	.000		.251**	.009	R41PR
R42DEV	.540**	.000		.170	.080	R42DEV
R43PR	.553**	.000		.199*	.040	R43PR
R44ENG	.468**	.000		.269**	.005	R44ENG
R45ENG	.504**	.000		.073	.453	R45ENG
R46ENG	.654**	.000		.135	.165	R46ENG
R47DEV	.428**	.000		.068	.489	R47DEV
R48ENG	.533**	.000		.035	.721	R48ENG
R49ENG	.624**	.000		.249**	.010	R49ENG
R50DEV	.665**	.000		.075	.444	R50DEV
R51PR	.440**	.000		.125	.199	R51PR
R52ENG	.598**	.000		.142	.146	R52ENG
R53ENG	.654**	.000		.130	.183	R53ENG
R54DEV	.561**	.000		.059	.547	R54DEV
R55DEV	.594**	.000		.120	.217	R55DEV
R56DEV	.464**	.000		-.038	.700	R56DEV
R57ENG	.620**	.000		.211*	.029	R57ENG
R58ENG	.580**	.000		.092	.348	R58ENG

R59PR	.611**	.000		.062	.525	R59PR
R60DEV	.557**	.000		.080	.414	R60DEV
R61ENG	.655**	.000		.151	.119	R61ENG
R62ENG	.571**	.000		.044	.656	R62ENG
R63PR	.461**	.000		-.073	.457	R63PR
R64DEV	.689**	.000		.079	.417	R64DEV
Risk-to-quality	1			.285**	.003	Risk-to-quality
S4DEV	.285**	.003		1		S4DEV

11.5.1 *Risk influence on QUALITY*

In this section, the correlation between the interacted factors in our network is examined to find the significance of the relationship in order to determine the strongest and weakest relationships among them.

From the table, we can see the relationship between factors in the risk to QUALITY network factors was investigated using Pearson product-moment correlation coefficient. The results show that 24 factors are significantly correlated with S4DEV; for 12 of these, the correlation is significant at the 0.01 level and for the other 12 it is significant at the 0.05 level. There was a strong and positive correlation between risk QUALITY and S4DEV [r= 0.285, p<.0005]. This also supports the results that claim that S4DEV is the most central node and has a strong relationship in the network. R14DEV Inefficient team capabilities has the strongest and most positive relationship with S4DEV of all the risk factors [r= 0. 448, p<.0005]. These results emphasise the importance of team capabilities perception of project quality. Furthermore, S4DEV has a strong and positive correlation with R17PR [r= 0. 382, p<.0005].

R23DEV Project manager lacks experience has the third strongest relation with S4DEV; this focusses again on the importance of the project management's experience in relation to raising the effectiveness of the project management style. This research also noted that S4DEV has five factors with a negative relationship: R63PR, R26DEV, R37DEV, R56DEV and R6ENG. However, because of their insignificant values they have not been taken into account as either a strong or a weak relationship.

The results also show that there was a strong, positive correlation between risk QUALITY and R34ENG Incompatible development environment [r= 0. 697, p<.0005], with high levels of incompatible development environment correlating with high levels of project QUALITY. R64DEV Project distribution also has a positive and strong relationship with project QUALITY [r= 0. 689, p<.0005]. It interesting to see project distribution having a strong relationship with quality as the reason behind that could because it has been recognised by many project stakeholders as a controlling project factor as "The coding and the other parts of the design and testing are often done offshore" (Jager et al., 2008). It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised to have a correlation that is significant at the 0.01 level with the risk-to-QUALITY.

11.5.2 *Success factors influencing QUALITY*

Table 11-6 R17PR and Risk-to-quality association network

Factors	SUCCESS-TO-QUALITY			R17PR		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
S1ENG	.632**	.000		.260**	.007	S1ENG
S2ENG	.701**	.000		.155	.111	S2ENG
S3DEV	.566**	.000		.113	.248	S3DEV
S4DEV	.615**	.000		.382**	.000	S4DEV
S5DEV	.570**	.000		.299**	.002	S5DEV
S6PR	.564**	.000		.155	.111	S6PR
S7DEV	.624**	.000		.181	.062	S7DEV
S8PR	.670**	.000		.422**	.000	S8PR
S9DEV	.546**	.000		.036	.714	S9DEV
S10ENG	.594**	.000		.290**	.002	S10ENG
S11DEV	.633**	.000		.092	.346	S11DEV
S12DEV	.686**	.000		.203*	.036	S12DEV
S13DEV	.749**	.000		.275**	.004	S13DEV
S14DEV	.577**	.000		.176	.070	S14DEV
S15PR	.764**	.000		.326**	.001	S15PR
S16DEV	.704**	.000		.255**	.008	S16DEV
S17DEV	.632**	.000		.190	.050	S17DEV
S18DEV	.710**	.000		.263**	.006	S18DEV
S19PR	.601**	.000		.166	.087	S19PR

S20DEV	.689**	.000		.107	.272	S20DEV
S21DEV	.558**	.000		.096	.324	S21DEV
S22ENG	.555**	.000		.276**	.004	S22ENG
S23PR	.619**	.000		.181	.062	S23PR
S24DEV	.761**	.000		.272**	.005	S24DEV
S25DEV	.675**	.000		.061	.530	S25DEV
S26DEV	.542**	.000		.042	.666	S26DEV
S27PR	.570**	.000		.180	.063	S27PR
S28PR	.746**	.000		.221*	.022	S28PR
S29DEV	.608**	.000		.243*	.012	S29DEV
S30PR	.584**	.000		.167	.086	S30PR
S31DEV	.702**	.000		.220*	.023	S31DEV
S32DEV	.694**	.000		.185	.056	S32DEV
S33DEV	.561**	.000		.182	.061	S33DEV
S34ENG	.543**	.000		.138	.157	S34ENG
S35ENG	.496**	.000		.166	.088	S35ENG
S37DEV	.568**	.000		.209*	.031	S37DEV
S38DEV	.629**	.000		.351**	.000	S38DEV
Success-to-quality	1			.331**	.001	Success-to-quality
R17PR	.331**	.001		1		R17PR

This section examines the correlation between the interacted factors in the success-to-QUALITY network to find the significance of the relationship in order to determine the direction of the relationship, and the strong and weak relationships among them.

The results shows that 18 factors are significantly correlated with R17PR. Thirteen of these have a correlation that is significant at the 0.01 level and five have a correlation that is significant at the 0.05 level. There was a strong, positive correlation between R17PR Resource insufficiency and S8PR Realistic budget [r= 0. 422, p<.0005]. Also, it is noted that S8PR has a stronger and more positive relationship with SUCCESS-TO-QUALITY [r= 0. 670, p<.0005] but it is not in the top five success factors correlating with success to cost . S4DEV has the second strongest and most positive relationship with R17PR [r= 0. 382, p<.0005]. Furthermore, success-to-quality has the fifth strongest and most positive relationship with R17PR [r= 0. 0.331, p<.0005], which emphasises the importance of this factor in relation to quality also as the controlling factor in the network. S12DEV Proper planning has the weakest significant relationship with R17PR [r= 0. 203, p<.0005].

The results also show that there was a strong, positive correlation between SUCCESS-TO-QUALITY and S15PR Adequate resources [r= 0. 764, p<.0005]. S24DEV Clear assignment of roles and responsibilities has the second strongest relationship with Success-to-quality [r= 0. 761, p<.0005]; this factor is also important, as has been mentioned, as it makes the most contribution. It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised to have a correlation that is significant at the 0.01 level with SUCCESS-TO-QUALITY , which agrees with this research's claim of their importance to what have been mentioned as important to the QUALITY ego network as well as their interaction with R17PR. Although R17PR has the lowest relationship strength, that could be, as has been mentioned, because the network was about the influence of the most central risk factor whilst all the factors have been correlated from a success point of view to success-to-QUALITY; therefore, giving R17PR a huge but indirect relationship with success-to-QUALITY.

11.6 **Summary**

This chapter has analysed the quality ego network characteristics, and identified the top central factors. Another objective achieved in this chapter was to investigate the contribution of risk and success factors to software project quality. In addition, the relationships between the factors have been also explored and analysed.

Chapter 12: Time ego network

Chapter 12: Time ego network

12.1 Introduction

This chapter explores the relationship and contribution of risk and success factors for the time ego network. It will analyse the time ego network top central factors, then define and isolate the controlling factors from the risk and success factors, and then analyse and explore the association among the factors.

12.2 Time ego network

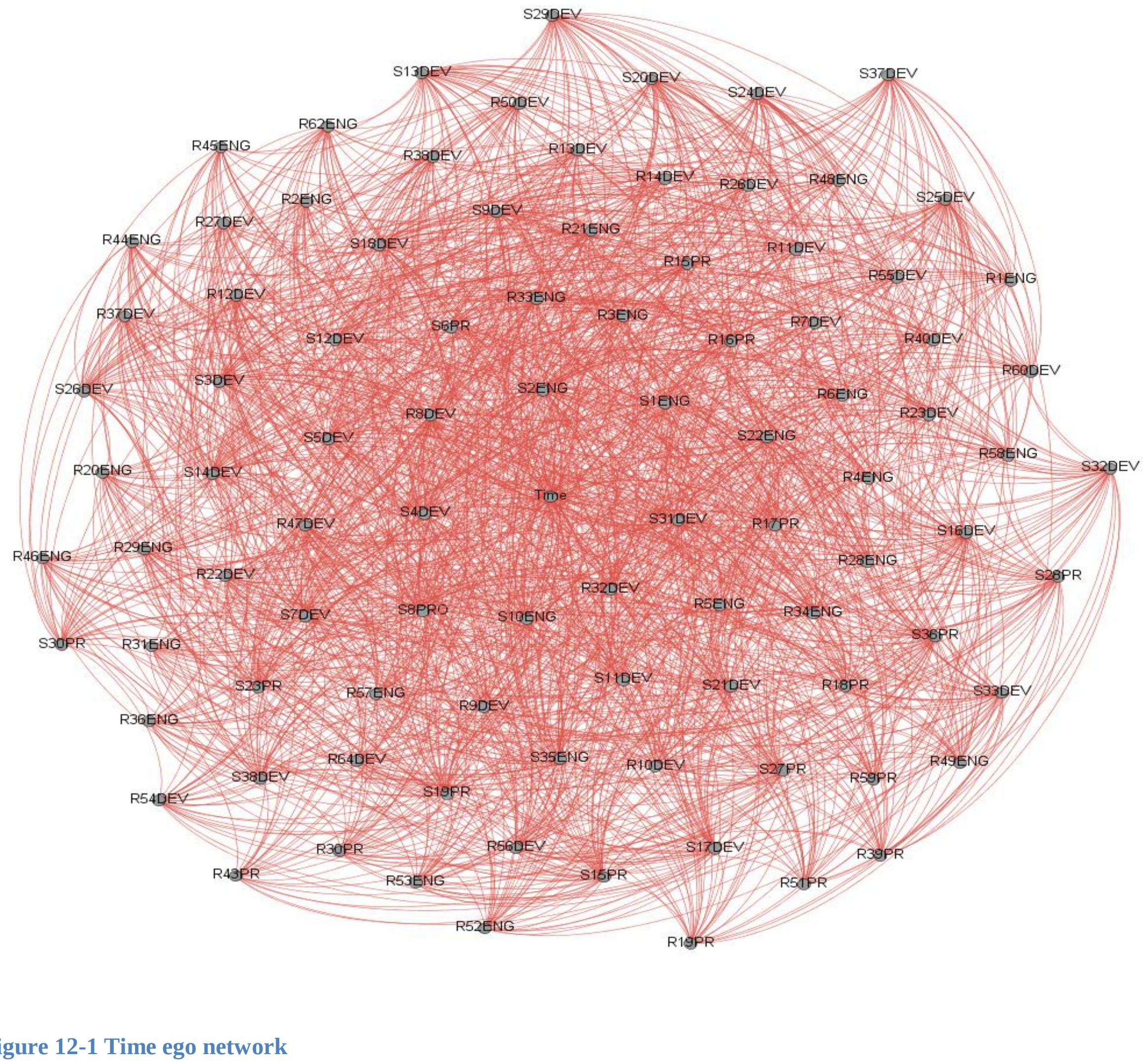


Figure 12-1 Time ego network

In this network, 95 factors are connected directly to TIME out of a maximum of 102 possible factor connections. Seven factors are not directly connected to time. The other three criteria have been isolated and removed from the network to study the interaction of the risk and success factors in relation to time without the influence of the other success criteria, in order to obtain the maximum interaction impact for time itself.

12.2.1 *Time top five central success factors*

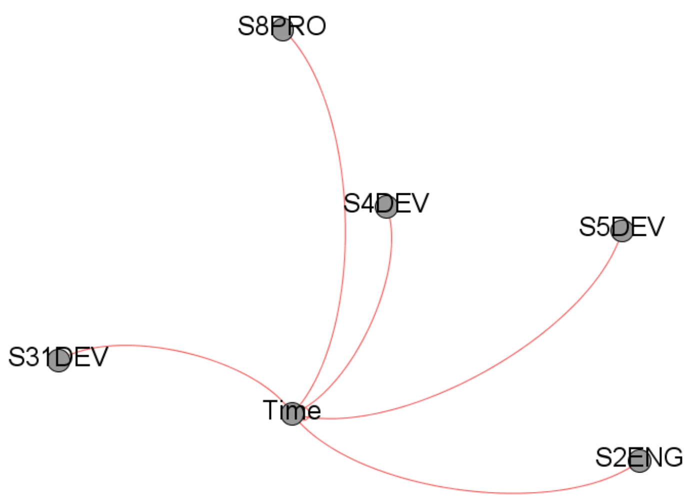


Table 12-1 Top five central success factors in Time ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
S4DEV	58	1.382979	125.5613	0.580933	1
S8PR	56	1.404255	116.2271	0.562735	2
S31DEV	56	1.404255	115.6634	0.563748	3
S2ENG	55	1.414894	112.0356	0.552576	4
S5DEV	55	1.414894	110.806	0.555182	5

S4DEV Efficient project management is the most central factor in terms of degree to the time ego network with a degree of 58 connections. Efficient project management is directly connected to the largest number of factors. Thus, this factor is considered to be the central one in terms of direct impact on time saving or, in other words, its impact can spread directly to the majority of network risk and success factors. That means that anything that affects efficient project management will have a huge impact on time saving or a software project running over time.

Figure 12-2 Time ego network with a degree of 58

Although S8PR Realistic budget and S31DEV Organisational culture have a degree centrality of 56 connections and closeness centrality value of 1.40, S8PR Realistic budget is more central than S31DEV Organisational culture support because it has the shortest paths with a betweenness value of 116.28. This indicates that it has more control in the network than S31DEV, which has a lower betweenness value of 115.66.

Figure 12-3 Time ego network with a degree of 56

Although S2ENG Clear objectives and goals and S5DEV Top level management support have a degree centrality of 55 connections and a closeness centrality value of 1.41, the former is more central than the latter because it has more shortest paths with a betweenness value of 112.06. This indicates that it has more control in the network than S31DEV, which has a lower betweenness value of 110.81.

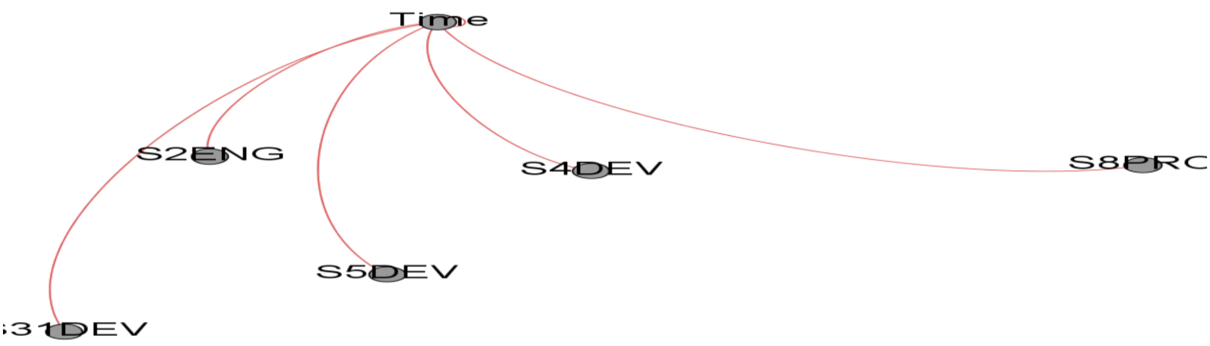


Figure 12-4 Time ego network with a degree of 55

12.2.2 *TIME degree centrality top five Risk factors*

Table 12-2 Top five central Risk factors in Time ego network

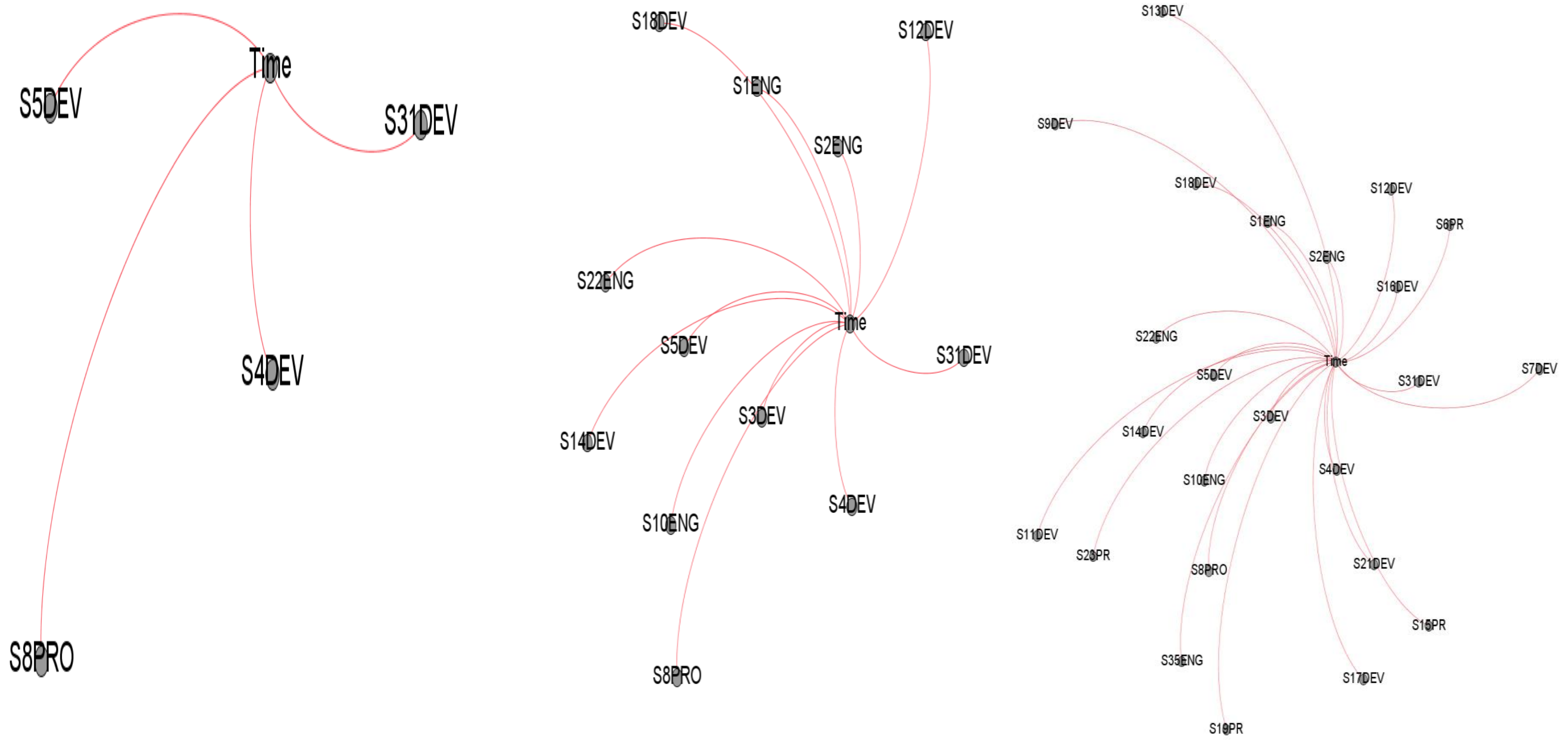
Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
R17PR	38	1.595745	35.51328	0.468335	1
R16PR	37	1.606383	33.15496	0.458533	2
R23DEV	36	1.617021	31.01978	0.44779	3
R11DEV	35	1.62766	29.61487	0.434096	4
R18PR	35	1.62766	29.24034	0.436476	5
R3ENG	35	1.62766	28.79576	0.438323	6
R15PR	35	1.62766	28.79576	0.438323	7

Resource insufficiency is considered as the most central factor to the time ego network as it has a direct impact on 38 success factors and the lowest average distance to all the other factors, with a closeness value of 1.60. Likewise, it has the highest number of shortest paths to all other factors in the network with a betweenness value of 35.51, which indicates that it has more control in the time ego network.

R16PR is considered the second most central factor to the time network as it has a direct impact on 37 success factors and the lowest average distance to all the factors, with a closeness value of 1.61. Furthermore, it has a betweenness value of 33.15. R23DEV Lack of experience of project manager is the third most central factor in the quality ego network with a degree of 36 connections and closeness centrality value of 1.61. Although R11DEV, R18PR, R3ENG and R15PR have the same centrality value in terms of the number of connections to all success factors as well as their distance from all other factors in the time ego network, R11DEV Unrealistic resource planning is considered the fourth central factor in the time ego network as it has the highest number of shortest paths due to its betweenness value of 29.61. R18PR User resistance is the fifth central factor in the time ego network as it has a betweenness value of 29.24

12.3 Isolating the risk and success factors in the TIME ego network

The three-step methodology has been applied in this section, as has been explained previously. The aim of this section is to find the factors controlling the influence of risk factors over the success factors.

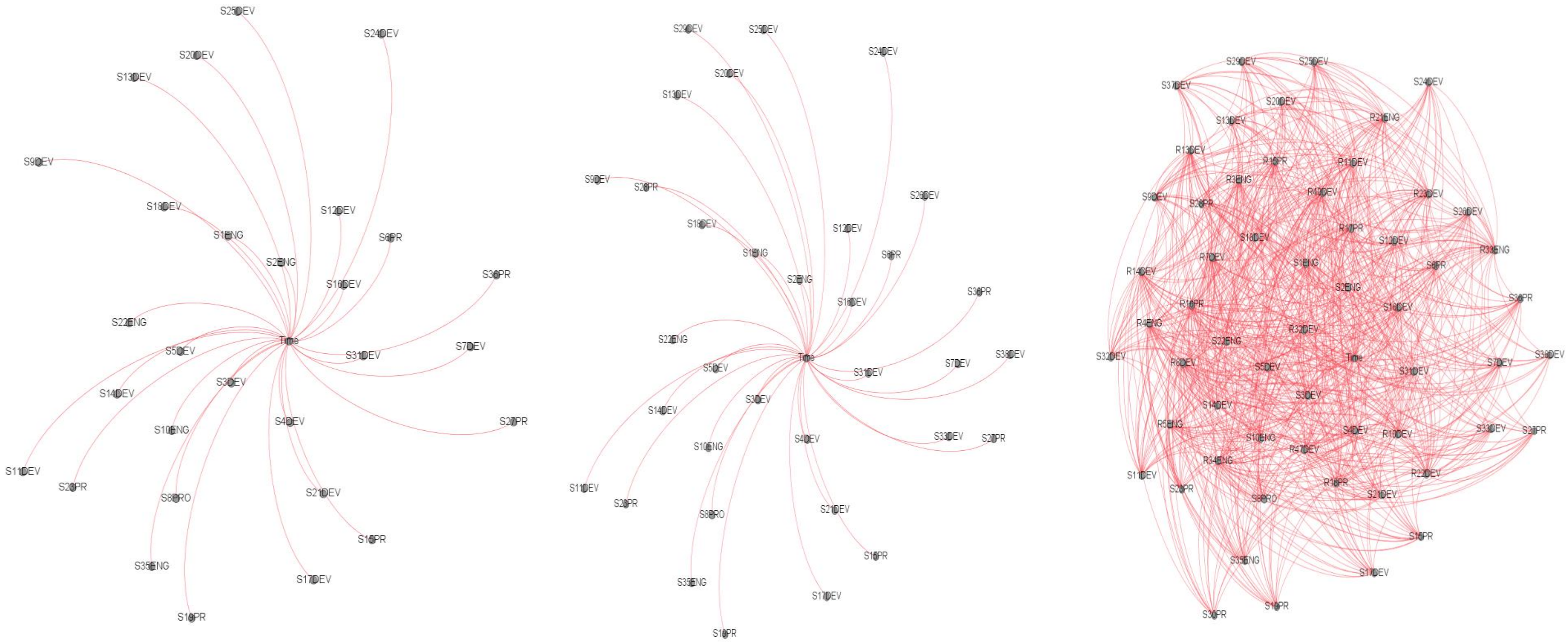


This graph shows the success factors with a degree of 63 and more in the time ego network

This graph shows the success factors with a degree of 57 and more in the time ego network

This graph shows the success factors with a degree of 52 and more in the time ego network

Figure 12-5 The change that occurs in the time ego network and its success factors when adjusting the degree topology value part one



This graph shows the success factors with a degree of 47 and more in the time ego network

This graph shows the success factors with a degree of 42 and more in the time ego network

This graph shows the success factors with a degree of 34 and more in the time ego network

Figure 12-6 The change that occurs in the time ego network and its success factors when adjusting the degree topology value part two

12.3.1 *Risk factors influencing TIME*

The algorithm has been applied in various steps. In the degree range of 64 it is noticeable that the most central factors are S4DEV, S8PR, S5DEV and S31DEV. When reducing the minimum degree range to 57, the number of factors connected to TIME increases to 12. In the minimum degree range of 52, there are 24 factors connected to the TIME ego network. Twenty-nine factors are respectively central to TIME when the degree of 47 is applied. As can be seen above, there are 34 factors with a degree of 42 or higher. The last degree range applied in the TIME ego network is 34, with 58 factors interacting with TIME directly.

12.3.1.1 *The success factor that has the most control and influence over the risk factors time ego network*

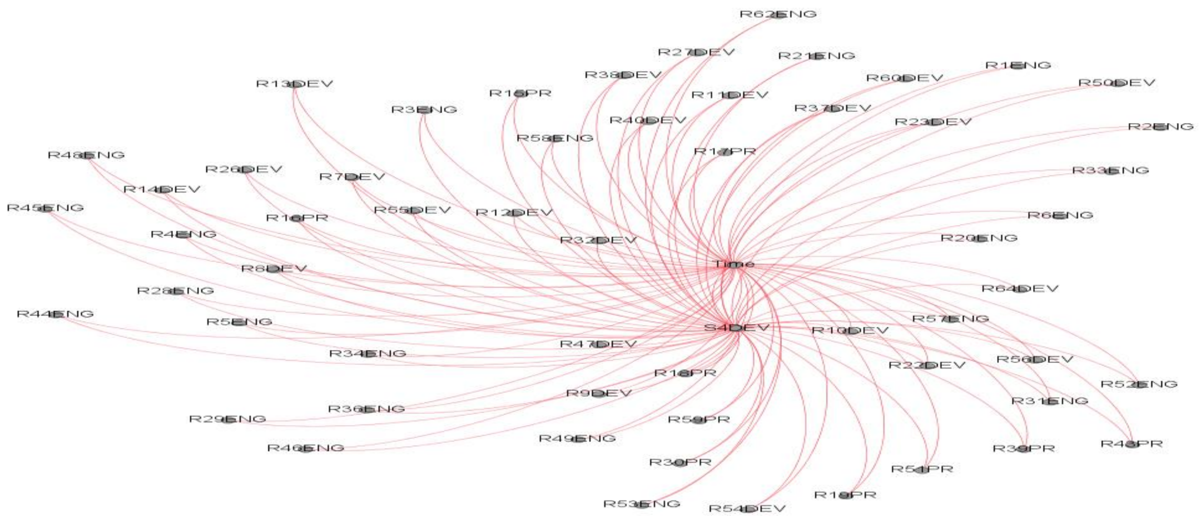
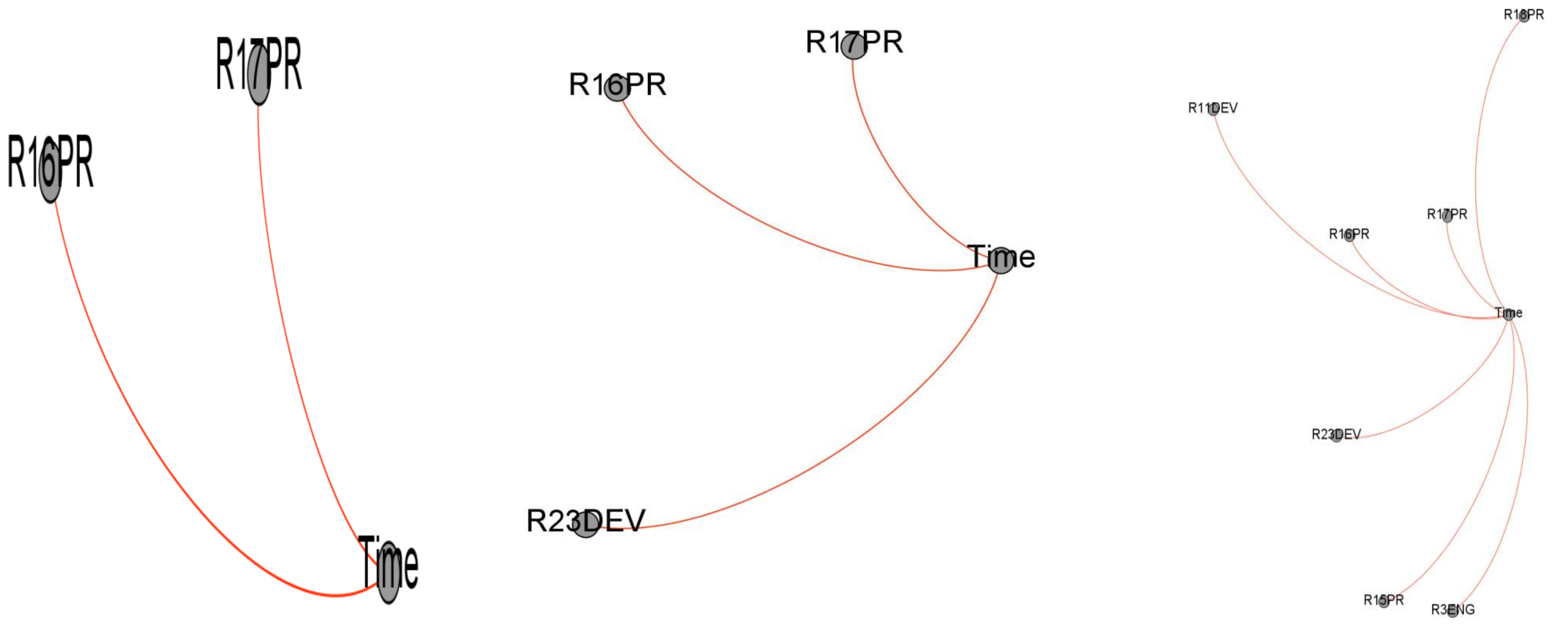


Figure 12-7S4DEV's influence on risk factors in the TIME ego network

It is noticeable that the most central node in terms of its betweenness is S4DEV Effective project management,. S4DEV is also the most central factor in the network in terms of its betweenness value of 125.57 . Furthermore, S4DEV is connected to 57 risk factors connected to the TIME ego network which means, in other words, that Effective project management has control and influence on all of the risk factors connecting to the TIME ego network. It should also be noted that this factor is in close proximity to the rest of the risk factors; this is supported by its Closeness centrality value of 1.38.

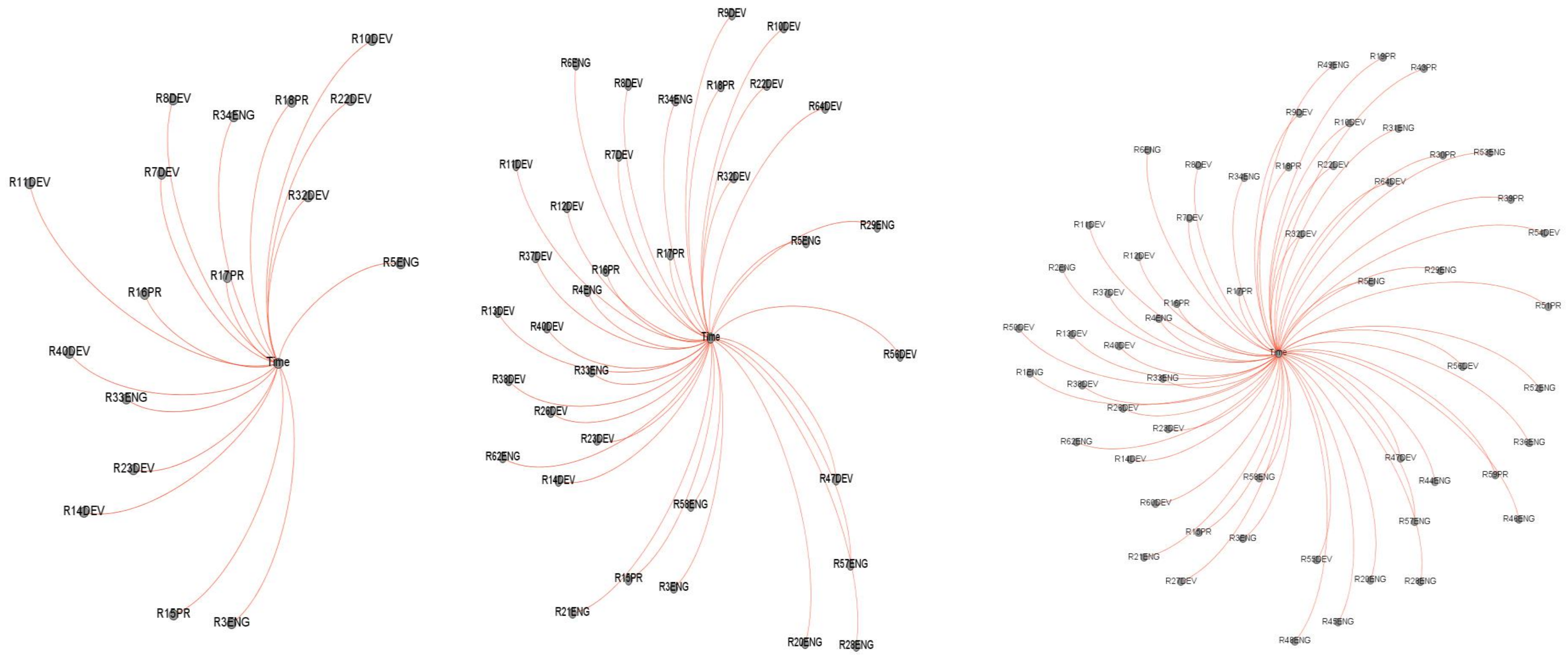


This graph shows the risk factors with a degree of 38 and more in the time ego network

This graph shows the risk factors with a degree of 36 and more in the time ego network

This graph shows the risk factors with a degree of 35 and more in the time ego network

Figure 12-8 The change that occurs in the time ego network and its risk factors when adjusting the degree topology value part one



This graph shows the risk factors with a degree of 34 and more in the time ego network

This graph shows the risk factors with a degree of 30 and more in the time ego network

This graph shows the risk factors with a degree of 25 and more in the time ego network

Figure 12-9 The change that occurs in the time ego network and its risk factors when adjusting the degree topology value part two

12.3.2 *Success factors influencing TIME*

The algorithm has been applied in various steps. In the degree range of 37, it can be seen from the graph above that there are two factors connected to TIME, R16PR Unrealistic budget and R17PR Resource insufficiency. When reducing the minimum degree range to 36 the number of factors connected to TIME increases to three. In the minimum degree range of 35, seven factors are connected to the TIME ego network. Seventeen factors are respectively connected to TIME when the degree of 34 is applied. As can be seen above, there are 35 factors with a degree of 30 or higher. The last degree range applied in the TIME ego network is in the minimum degree range of 25, with 57 factors interacting with TIME directly.

12.3.2.1 *The risk factor that has the most control and influence over the success factors in time ego network*

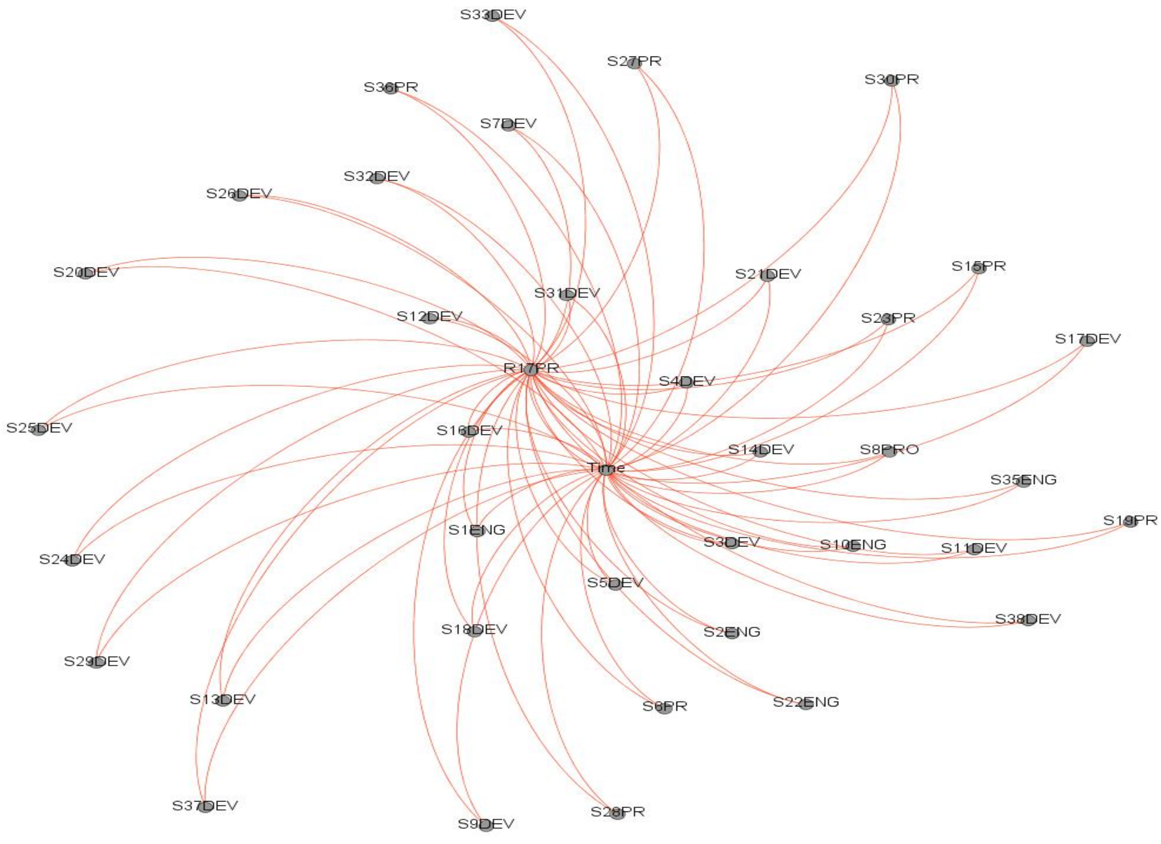


Figure 12-10R43PR's influence on success factors in the TIME ego network

As mentioned above, the betweenness centrality will be used to determine the most central risk factor and-find how it influences the TIME network as well as exploring the interaction in its isolated network. It is noticeable that the most central node is R17PR Resource inefficiency. Furthermore, R17PR is connected to 37 success factors connected in the ego risk network but is not connected to S36PR Commitment of stakeholders. R17PR is the most central factor in terms of closeness in the TIME ego network. It should also be noted that this factor is in close proximity to the rest of the success factors; this is supported by its Closeness centrality value of 1.62, which reflects the spread time and speed of this factor's impact on the other factors. Modelling of the relationship between the isolated network nodes

Models

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to TIME	.998[a]	.996	.992	.03645	.996	218.732	58	47	.000
Success to TIME	1.000[a]	.999	.999	.01545	.999	2620.123	38	68	.000

12.3.3 *Risk factors association model results*

The Risk to TIME sub-model is associated with a large number of risk impacts. There is an 99% association between Risk to TIME and their risk impact events, which reflects the influence of S4DEV on the risk factors' interaction with success. The p-values shown above determine how good a fit the model is for the observed data. The p-values that are ≤ 5% mean that the null hypothesis can be rejected, as it can be seen that the p-values in Risk to TIME are 000% ≤ 5%.

Table 12-3 Risk factors association model results for Risk-to-time

Model	Unstandardised Coefficients			Model	Unstandardised Coefficients		
	B	Std. Error	Sig.		B	Std. Error	Sig.
(Constant)	-.0406	.055	.466	R32DEV	.0159	.009	.091
R1ENG	.0241	.010	.023	R33ENG	.0088	.009	.334
R2ENG	.0171	.008	.036	R34ENG	.0186	.009	.035
R3ENG	.0110	.010	.258	R36ENG	.0086	.009	.356
R4ENG	.0287	.008	.001	R37DEV	.0245	.008	.002
R5ENG	.0018	.010	.856	R38DEV	.0091	.009	.306
R6ENG	.0192	.008	.018	R39PR	.0052	.009	.551
R7DEV	.0169	.010	.083	R40DEV	.0207	.009	.028
R8DEV	.0244	.008	.003	R43PR	.0208	.007	.003
R9DEV	.0354	.009	.000	R44ENG	.0267	.012	.026
R10DEV	-.0011	.010	.909	R45ENG	.0057	.010	.568
R11DEV	.0239	.010	.017	R46ENG	.0248	.011	.032
R12DEV	.0099	.008	.244	R47DEV	.0081	.008	.340
R13DEV	.0226	.008	.010	R48ENG	.0167	.009	.060
R14DEV	.0149	.010	.157	R49ENG	.0143	.007	.047
R15PR	.0253	.011	.024	R50DEV	.0123	.009	.198
R16PR	.0300	.010	.004	R51PR	.0074	.009	.426
R17PR	.0101	.010	.297	R52ENG	.0125	.008	.133
R18PR	.0155	.007	.026	R53ENG	.0114	.010	.247
R19PR	.0226	.006	.001	R54DEV	.0200	.011	.077
R20ENG	.0164	.009	.079	R55DEV	.0244	.008	.005

R21ENG	.0181	.011	.094	R56DEV	.0281	.008	.002
R22DEV	.0244	.009	.012	R57ENG	.0167	.009	.059
R23DEV	.0256	.011	.022	R58ENG	.0035	.009	.691
R26DEV	.0217	.006	.001	R59PR	.0129	.010	.223
R27DEV	.0267	.009	.007	R60DEV	.0152	.008	.064
R28ENG	.0131	.009	.150	R62ENG	.0285	.010	.008
R29ENG	.0150	.008	.069	R64DEV	.0289	.007	.000
R30PR	.0206	.008	.012	S4DEV	-.0092	.009	.283
R31ENG	.0333	.007	.000				
	a. Dependent Variable: RISK-TO-TIME						

It noticeable that R9DEV Inadequate infrastructure has the largest beta coefficient of 0.0354, which gives this factor the strongest unique contribution to explaining the project RISK-TO-TIME . For every unit increase in R9DEV a 0.0354 unit increase in TIME is predicted. Its impact on Risk-to-time could be because it usually results in conflict among team members (Silva et al., 2015) which delays the project process time. Also, another reason could be because there are a large number of elements related to infrastructure that have an impact on project duration; for example, if there is no reliable power supply or alternative power source, it would be impossible to develop a software product on time. Also, the Beta value for R31ENG, R16PR and R64DEV was slightly lower than R9DEV, indicating that the latter made less of a contribution to Risk-to-time .

12.3.4 *Success factors association model results*

Table 12-4 Success factors association model results for success-to-time

Model	Unstandardised Coefficients		Sig.	Model	Unstandardised Coefficients		Sig.
	B	Std. Error			B	Std. Error	
(Constant)	.005	.015	.762	S20DEV	.028	.004	.000
S1ENG	.023	.004	.000	S21DEV	.027	.004	.000
S2ENG	.030	.004	.000	S22ENG	.033	.004	.000
S3DEV	.029	.005	.000	S23PR	.028	.003	.000
S4DEV	.021	.004	.000	S24DEV	.026	.004	.000
S5DEV	.028	.003	.000	S25DEV	.024	.003	.000
S6PR	.027	.003	.000	S26DEV	.025	.003	.000
S7DEV	.027	.003	.000	S27PR	.027	.003	.000
S8PR	.023	.004	.000	S28PR	.030	.003	.000
S9DEV	.028	.004	.000	S29DEV	.029	.004	.000
S10ENG	.023	.004	.000	S30PR	.025	.003	.000
S11DEV	.022	.004	.000	S31DEV	.030	.003	.000
S12DEV	.021	.004	.000	S32DEV	.033	.003	.000
S13DEV	.034	.003	.000	S33DEV	.036	.004	.000
S14DEV	.033	.004	.000	S35ENG	.032	.004	.000
S15PR	.018	.004	.000	S36PR	.034	.004	.000
S16DEV	.020	.004	.000	S37DEV	.024	.004	.000
S17DEV	.029	.003	.000	S38DEV	.026	.004	.000
S18DEV	.016	.004	.000	R17PR	0.00	.003	.986
S19PR	.031	.004	.000				
	a. Dependent Variable: SUCCESS-TO-TIME						

From Table 12-4 above it seems that S33DEV Project criticality makes the greatest contribution to project TIME with a value of .036. This result shows the importance of dealing with project criticality and how it affects the project delivery time, as an increase in unit of S33DEV has a positive impact on project TIME with an increase in value of 0.036. It also worth mentioning that project criticality is usually linked with software projects where agile methodology is not the best methodology to use (Sheffield & Lemétayer, 2013). It is interesting to note that S13DEV Up-to-date progress report and S36PR Commitment of stakeholders have relatively the second strongest contribution to SUCCESS-TO-TIME as it shows that up-to-date progress reports and step-by-step monitoring and involvement of the stakeholders undoubtedly have a positive effect on the duration of a project and result in the project being delivered on time. It is noticeable that all the success factors have a positive contribution to SUCCESS-TO-TIME .

12.4 The association between the network nodes

Table 12-5 S4DEV and Risk-to-time association network

Factors	RISK-TO-TIME			S4DEV		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
R1ENG	.083	.394		.086	.378	R1ENG
R2ENG	.477**	.000		.088	.366	R2ENG
R3ENG	.431**	.000		.146	.134	R3ENG
R4ENG	.377**	.000		.095	.328	R4ENG
R5ENG	.428**	.000		.097	.319	R5ENG
R6ENG	.326**	.001		.205*	.034	R6ENG
R7DEV	.461**	.000		.081	.409	R7DEV
R8DEV	.447**	.000		-.031	.749	R8DEV
R9DEV	.469**	.000		.120	.219	R9DEV
R10DEV	.295**	.002		.040	.680	R10DEV
R11DEV	.467**	.000		.186	.055	R11DEV
R12DEV	.487**	.000		.267**	.005	R12DEV
R13DEV	.414**	.000		.317**	.001	R13DEV
R14DEV	.467**	.000		.330**	.001	R14DEV
R15PR	.293**	.002		.111	.253	R15PR
R16PR	.448**	.000		.155	.110	R16PR
R17PR	.567**	.000		.128	.188	R17PR
R18PR	.478**	.000		.121	.213	R18PR
R19PR	.360**	.000		.010	.916	R19PR

R20ENG	.454**	.000		.055	.577	R20ENG
R21ENG	.476**	.000		.360**	.000	R21ENG
R22DEV	.339**	.000		.201*	.038	R22DEV
R23DEV	.406**	.000		.455**	.000	R23DEV
R26DEV	.295**	.002		.046	.635	R26DEV
R27DEV	.443**	.000		.390**	.000	R27DEV
R28ENG	.576**	.000		.084	.388	R28ENG
R29ENG	.530**	.000		.282**	.003	R29ENG
R30PR	.582**	.000		.077	.434	R30PR
R31ENG	.553**	.000		.112	.249	R31ENG
R32DEV	.493**	.000		.011	.910	R32DEV
R33ENG	.390**	.000		.186	.055	R33ENG
R34ENG	.561**	.000		.131	.177	R34ENG
R36ENG	.565**	.000		.080	.412	R36ENG
R37DEV	.408**	.000		.168	.084	R37DEV
R38DEV	.578**	.000		.163	.093	R38DEV
R39PR	.367**	.000		.082	.400	R39PR
R40DEV	.349**	.000		.154	.114	R40DEV
R43PR	.557**	.000		.237*	.014	R43PR
R44ENG	.339**	.000		.124	.203	R44ENG
R45ENG	.432**	.000		.110	.260	R45ENG
R46ENG	.629**	.000		.126	.195	R46ENG
R47DEV	.467**	.000		.177	.068	R47DEV
R48ENG	.534**	.000		.128	.190	R48ENG

R49ENG	.625**	.000		.038	.694	R49ENG
R50DEV	.587**	.000		.115	.237	R50DEV
R51PR	.587**	.000		.104	.284	R51PR
R52ENG	.604**	.000		.091	.351	R52ENG
R53ENG	.618**	.000		.170	.080	R53ENG
R54DEV	.585**	.000		.141	.147	R54DEV
R55DEV	.596**	.000		-.083	.393	R55DEV
R56DEV	.576**	.000		.018	.857	R56DEV
R57ENG	.711**	.000		.203*	.036	R57ENG
R58ENG	.569**	.000		-.005	.956	R58ENG
R59PR	.627**	.000		.122	.209	R59PR
R60DEV	.677**	.000		.200*	.039	R60DEV
R62ENG	.613**	.000		.237*	.014	R62ENG
R64DEV	.605**	.000		.115	.236	R64DEV
Risk-to-time	1			.268**	.005	Risk-to-time
S4DEV	.268**	.005		1		S4DEV

12.4.1 *Risk influence on TIME*

From Table 12-5, the results show that 14 factors are significantly correlated with S4DEV. Eight have a correlation that is significant at the 0.01 level and six have a correlation that is significant at the 0.05 level. R23DEV Project manager lacks experience has the strongest and most positive relationship with S4DEV of all the risk factors [r= 0. 455, p<.0005]. These results emphasise the importance of managers' experience and how it is correlated with S4DEV. Furthermore, S27DEV E Extensive personnel hiring has the second strongest and most positive correlation with S4DEV [r= 0. 390, p<.0005].

This research also noted that S4DEV has three factors that have a negative relationship, R55DEV, R8DEV and R58ENG. However, because of their insignificant values they have not been taken into account as having a strong or a weak relationship.

The results also show that there was a strong, positive correlation between the risk TIME and R57ENG Use of cheap tools" [r= 0. 711, p<.0005], with high levels of use of cheap tools correlating with high levels project risk-to-time , which makes it the strongest relationship factor with risk-to-time . Although the use of cheap tools could help a project to save money, on the other hand it affects the project delivery time, as the developers have to spend time to modify the tools to suit the purpose of the project. For example, some projects tend to use cheap or free tools like free compilers instead of paying for full feature compilers (Koopman, 2010).

R60DEV No training for managing outsource relationships also has a positive and strong relationship with project RISK-TO-TIME [r= 0. 677, p<.0005]. It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised to have a correlation that is significant at the 0.01 level with the risk-to-TIME. The only factor that does not have a significant correlation with risk-to-time is R1ENG Unclear customer requirements. This research believes that this could be because unclear customer requirements will negatively affect the quality of a project as this factor has a correlation value of [r=0.292, p<.0005]. It is noted that the results of unclear customer requirements usually appear at the delivery of a project, especially if there were no customer involvement in the project duration. This is also linked to lack of skills to deal with outsourced project partners, R60DEV, which has been recognised as having the second strongest relationship with risk-to-time , as has been mentioned.

12.4.2 *Success factors' influence on TIME*

Table 12-6 R17PR and Success-to-time association network

Factors	SUCCESS-TO-TIME			R17PR		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
S1ENG	.596**	.000		.067	.494	S1ENG
S2ENG	.576**	.000		.170	.080	S2ENG
S3DEV	.539**	.000		.099	.312	S3DEV
S4DEV	.601**	.000		.128	.188	S4DEV
S5DEV	.544**	.000		.204*	.035	S5DEV
S6PR	.560**	.000		.216*	.025	S6PR
S7DEV	.521**	.000		.130	.182	S7DEV
S8PR	.651**	.000		.400**	.000	S8PR
S9DEV	.649**	.000		.310**	.001	S9DEV
S10ENG	.702**	.000		.300**	.002	S10ENG
S11DEV	.620**	.000		.164	.092	S11DEV
S12DEV	.659**	.000		.139	.153	S12DEV
S13DEV	.629**	.000		.324**	.001	S13DEV
S14DEV	.628**	.000		.345**	.000	S14DEV
S15PR	.580**	.000		.144	.138	S15PR
S16DEV	.724**	.000		.200*	.039	S16DEV
S17DEV	.609**	.000		.158	.104	S17DEV
S18DEV	.702**	.000		.356**	.000	S18DEV

S19PR	.666**	.000		.206*	.033	S19PR
S20DEV	.634**	.000		.181	.062	S20DEV
S21DEV	.577**	.000		.176	.070	S21DEV
S22ENG	.628**	.000		.256**	.008	S22ENG
S23PR	.632**	.000		.187	.053	S23PR
S24DEV	.658**	.000		.108	.268	S24DEV
S25DEV	.674**	.000		.290**	.002	S25DEV
S26DEV	.610**	.000		.313**	.001	S26DEV
S27PR	.639**	.000		.209*	.031	S27PR
S28PR	.578**	.000		.254**	.008	S28PR
S29DEV	.606**	.000		.257**	.008	S29DEV
S30PR	.585**	.000		.241*	.012	S30PR
S31DEV	.630**	.000		.146	.134	S31DEV
S32DEV	.603**	.000		.278**	.004	S32DEV
S33DEV	.675**	.000		.270**	.005	S33DEV
S35ENG	.609**	.000		.073	.452	S34ENG
S36PR	.592**	.000		.065	.503	S35ENG
S37DEV	.604**	.000		.131	.177	S37DEV
S38DEV	.636**	.000		.278**	.004	S38DEV
Success-to-time	1			.344**	.000	SUCCESS-TO-TIME
R17PR	.344**	.000		1		R17PR

The results show that 21 factors are significantly correlated with R17PR. Fifteen factors have a correlation that is significant at the 0.01 level and six have a correlation that is significant at the 0.05 level. There was a strong, positive correlation between R17PR Resource insufficiency and S8PR Realistic budget [r= 0. 400, p<.0005], which again, as in the cost to success and quality to success, shows the importance of this factor on the project success criteria. Also, it is noted that S8PR has a stronger and more positive relationship with SUCCESS-TO-TIME [r= 0. 344, p<.0005].

S18DEV Change management has the second strongest and most positive relationship with success to R17PR [r= 0. 356, p<.0005], which emphasises the importance of this factor to R17PR as it shows the importance of adopting change in a software development project as well as how it correlates with lack of resources.

The results also show that there was a strong, positive correlation between SUCCESS-TO-TIME and S16DEV Good leadership [r= 0. 724, p<.0005]. It is understandable that S10DEV Familiar with technology has the second strongest relationship with SUCCESS-TO-TIME [r= 0. 702, p<.0005] as familiarity with technology usually saves time whilst, if the technology used is not familiar to programmers, for example, this adds extra time to the project by designing prototypes to test and ensure that the technology has the capability to deliver what it is used for, especially in the most risky parts of the software (Zaied et al., 2013). It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised as having a correlation that is significant at the 0.01 level with the SUCCESS-TO-TIME , which agrees with this research’s claim about their importance to what have been mentioned as important to the TIME ego network as well as their interaction with R17PR. Although R17PR has the lowest relationship strength, that could be because, as has been mentioned, the network was about the influence of the most central risk factor where all the factors have been correlated from a success point of view to success-to-TIME, which results in R17PR having a huge but indirect relationship with success-to-TIME.

12.5 Summary

This chapter has investigated and analysed the time ego network interdependency. It has identified the top central factors, analysed the contribution of the factors in predicting the software project time, and explored the relationships among the factors.

Chapter 13: Scope ego network

Chapter 13: Scope ego network

13.1 Introduction

In this chapter, the scope ego network is isolated from the other criteria. The most central factors will be identified. Then this research will analyse the success and risk factors' contribution in predicting the project scope. The chapter will end by analysing the association among the factors.

13.2 Scope ego network

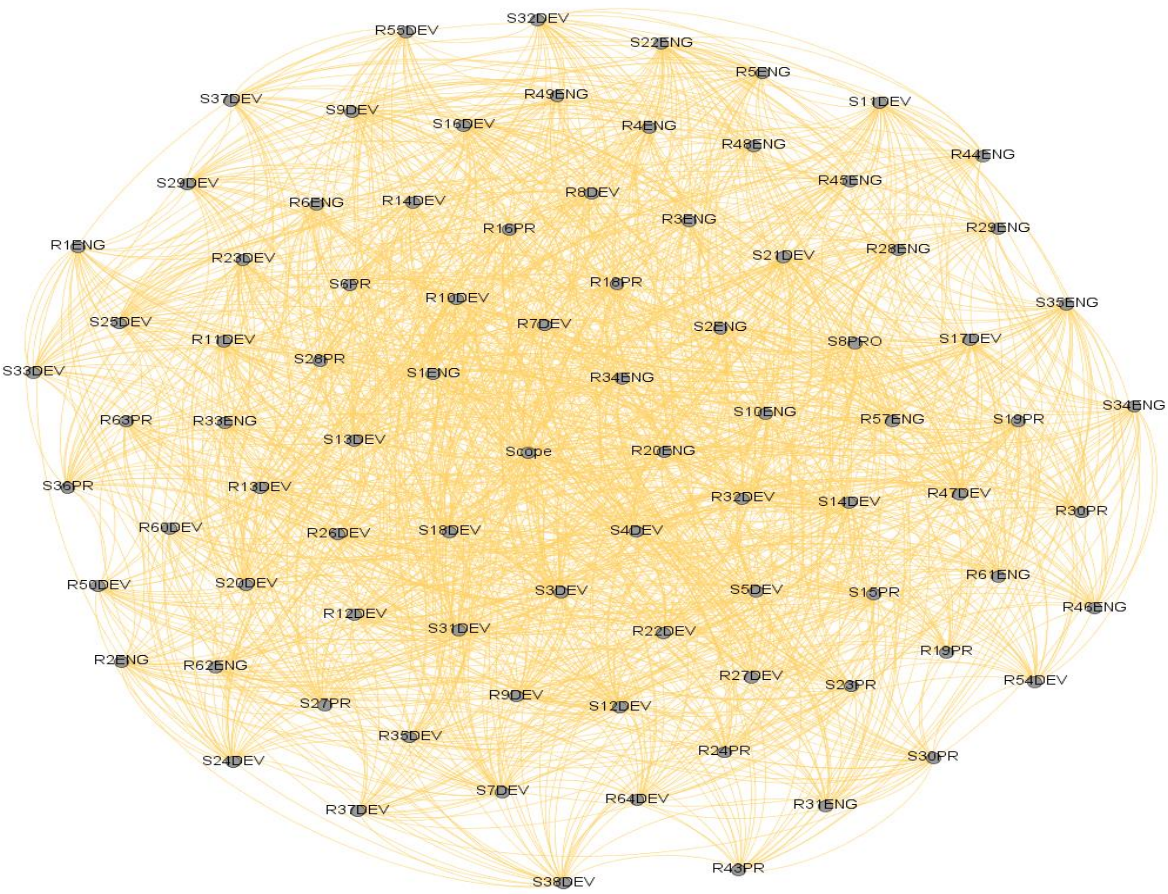


Figure 13-1 Scope ego network

In this network, 86 factors are directly connected to the Scope out of 103 possible factors. Seventeen factors are not directly connected to Scope, which makes it the lowest criterion in terms of number of factors directly connected when the ego network topology is applied. The other three criteria have been isolated and removed from the network to study the interactions of the risk and success factors in relation to Scope without any influence from these other success criteria, in order to obtain the maximum interaction impact on the Scope itself.

13.2.1 ***Scope top five central success factors***

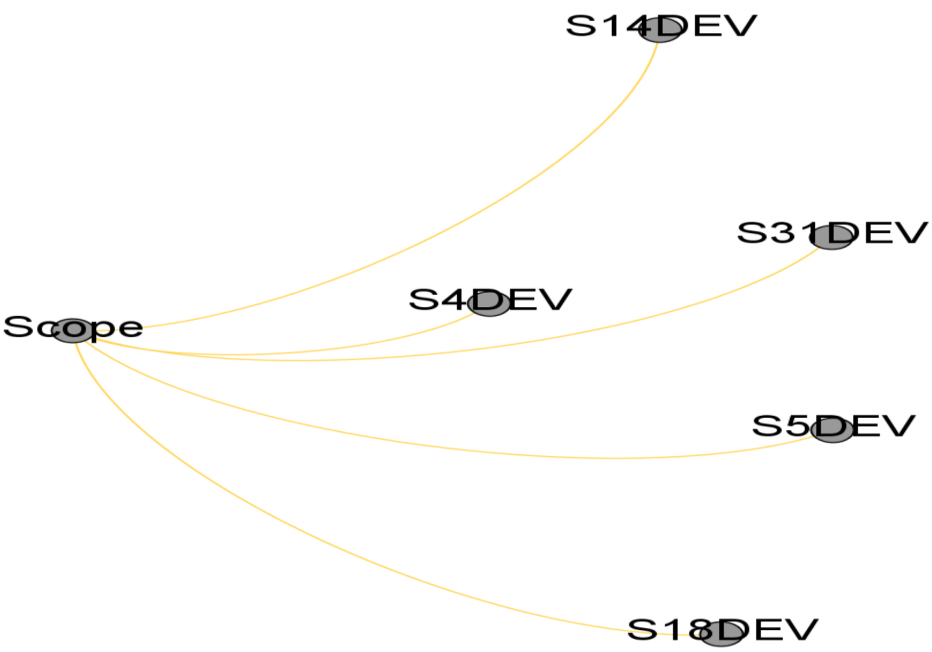


Figure 13-2 Top five central success factors

Table 13-1 Top five central success factors in Scope ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
S4DEV	49	1.423529	88.82	0.561479	1
S31DEV	48	1.435294	84.61	0.551658	2
S18DEV	47	1.447059	81.39	0.539474	3
S14DEV	47	1.447059	80.78	0.541148	4
S5DEV	47	1.447059	80.56	0.541694	5

S4DEV Efficient project management is the most central factor in terms of degree to the time ego network with a degree of 49 connections. It is directly connected to the largest number of factors. Therefore, this factor is considered to be the most central factor in terms of direct impact on Scope or, in other words, its impact can spread directly to the majority of network risk and success factors. Also, S4DEV has the minimum closeness value of 1.42, which also emphasises its centrality in this ego network, as it shows that S4DEV is the shortest distance from all other factors in the network. It is also worth mentioning that it has the average shortest path to all factors in the scope ego network with a betweenness value of 88.82.

S31DEV Organisational culture has a degree centrality of 48 connections and a closeness centrality value of 1.44, which makes it the second most central factor in the scope ego network. Although S18DEV, S14DEV and S5DEV have the same number of connections to other factors with a degree of 47 and also have the same distance from other factors with a betweenness value of 1.45, S18DEV Change management has the greatest number of shortest paths going through it with a betweenness value of 81.39, which makes it the third most central factor in this network compared to S14DEV, which has a betweenness value of 80.78, which makes it the fourth most central factor, and S5DEV is the fifth most central factor with a betweenness value of 80.56.

13.2.2 *Scope top five central risk factors*

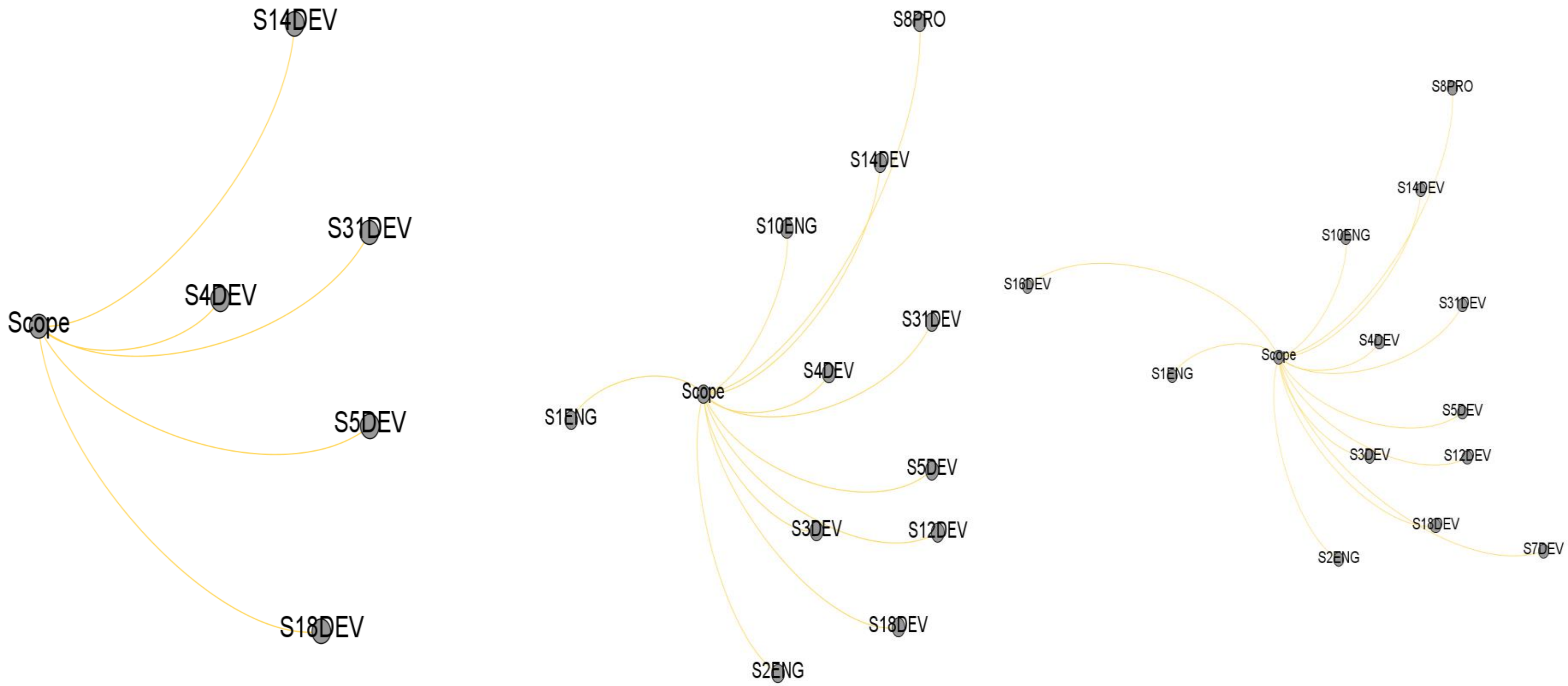
Table 13-2 Top five central Risk factors in Scope ego network

Id	Degree	Closeness Centrality	Betweenness Centrality	Eigenvector Centrality	Ranking
R16PR	37	1.564706	39.44	0.4818	1
R23DEV	36	1.576471	37.22	0.469388	2
R11DEV	35	1.588235	35.24	0.456105	3
R34ENG	35	1.588235	34.72	0.45816	4
R3ENG	35	1.588235	34.2	0.460933	5

R16PR Unrealistic budget is considered the most central factor in the scope network as it has a direct impact on 37 success factors and the lowest average distance from all the factors with a closeness value of 1.54. Furthermore, the betweenness value of 39.44. R23DEV Lack of experience of project manager is the second most central factor in the scope ego network with a degree of 36 connections and a closeness centrality value of 1.58. Also, it has the second highest number of shortest paths with a betweenness value of 37.22. Although R11DEV, R34ENG and R3ENG have the same centrality value in terms of their number of connections to all success factors, with 35 connections, as well as their distance from all other factors in the scope ego network, R11DEV Unrealistic resource planning is considered the third most central factor due to its betweenness value of 35.24 which shows that it has more average shortest paths than R34ENG Incompatible development environment with a betweenness value of 34.72, which makes it the fourth most central risk factor in this network. R3ENG is considered the fifth most central risk factor in this network as it has a betweenness value of 34.2 .

13.3 Isolating the risk and success factors in the SCOPE ego network

This section aims to isolate the risk factors that have the most betweenness value in the scope ego network in order to study the contribution of the factors in controlling prediction in the project scope. Furthermore, the relationship between the factors will also be explored.

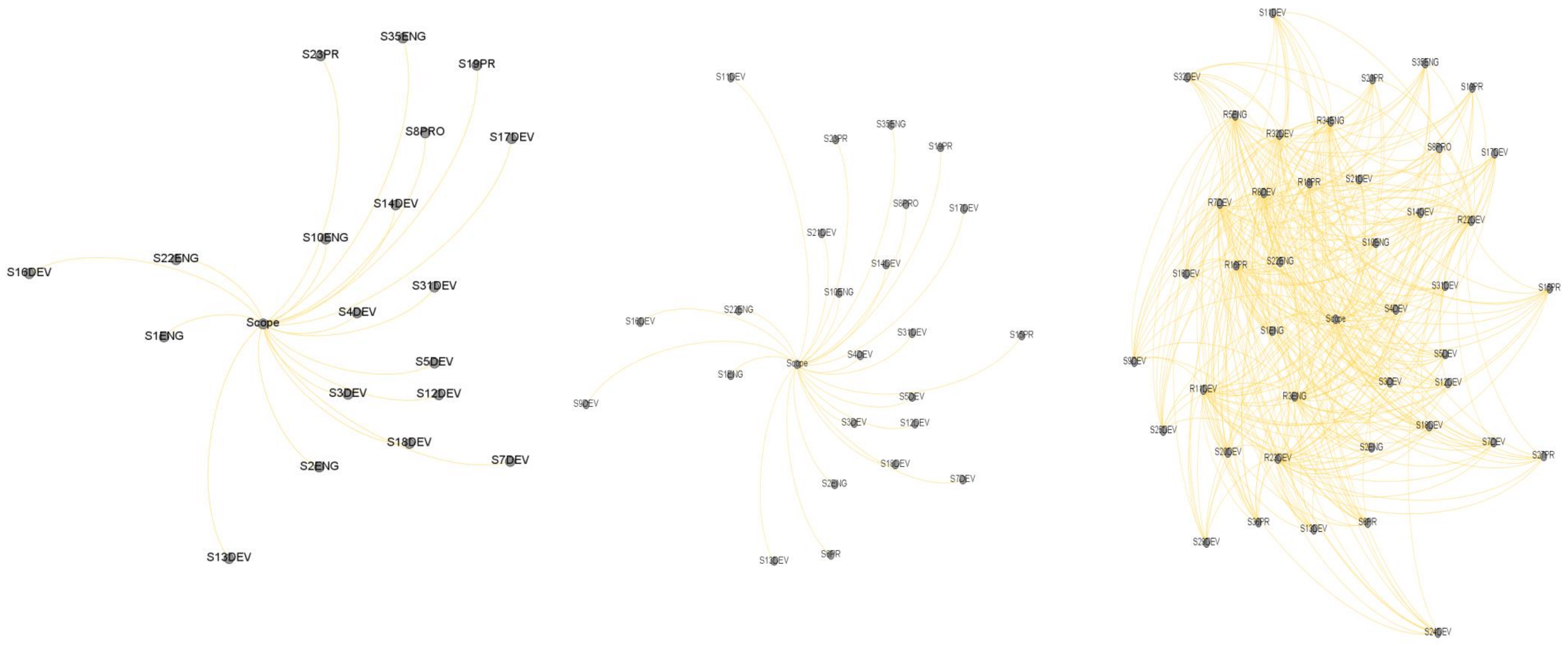


This graph shows the success factors with a degree of 47 and more in the scope ego network

This graph shows the success factors with a degree of 45 and more in the scope ego network

This graph shows the success factors with a degree of 42 and more in the scope ego network

Figure 13-3 The change that occurs in the scope ego network and its success factors when adjusting the degree topology value part one



This graph shows the success factors with a degree of 41 and more in the scope ego network

This graph shows the success factors with a degree of 39 and more in the scope ego network

This graph shows the success factors with a degree of 35 and more in the scope ego network

Figure 13-4 The change that occurs in the scope ego network and its success factors when adjusting the degree topology value part two

13.3.1 *Risk factors' influence on SCOPE*

In the degree range of 47, it is noticeable that the most central factors are S4DEV, S31DEV, S18DEV, S14DEV and S5DEV. When reducing the minimum degree range to 45, the number of factors connected to SCOPE increases to 11. In the minimum degree range of 42, 13 factors are connected to the SCOPE ego network. Nineteen factors are respectively central to SCOPE when the degree of 41 is applied. As can be seen above, there are 24 factors with a degree of 39 or higher. The last degree range applied in the SCOPE ego network is 34, with 42 factors interacting with SCOPE directly.

13.3.1.1 ***The success factor that has the most control and influence over the risk factors in scope ego network***

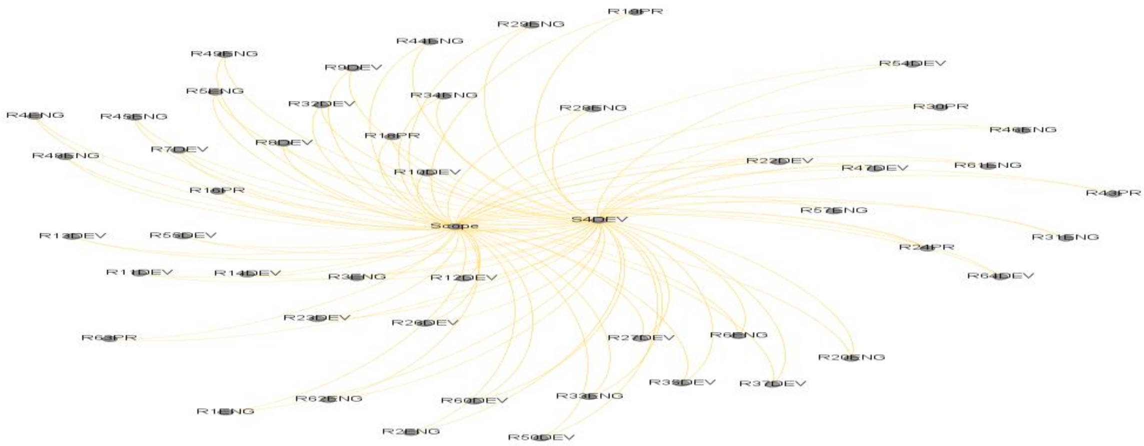
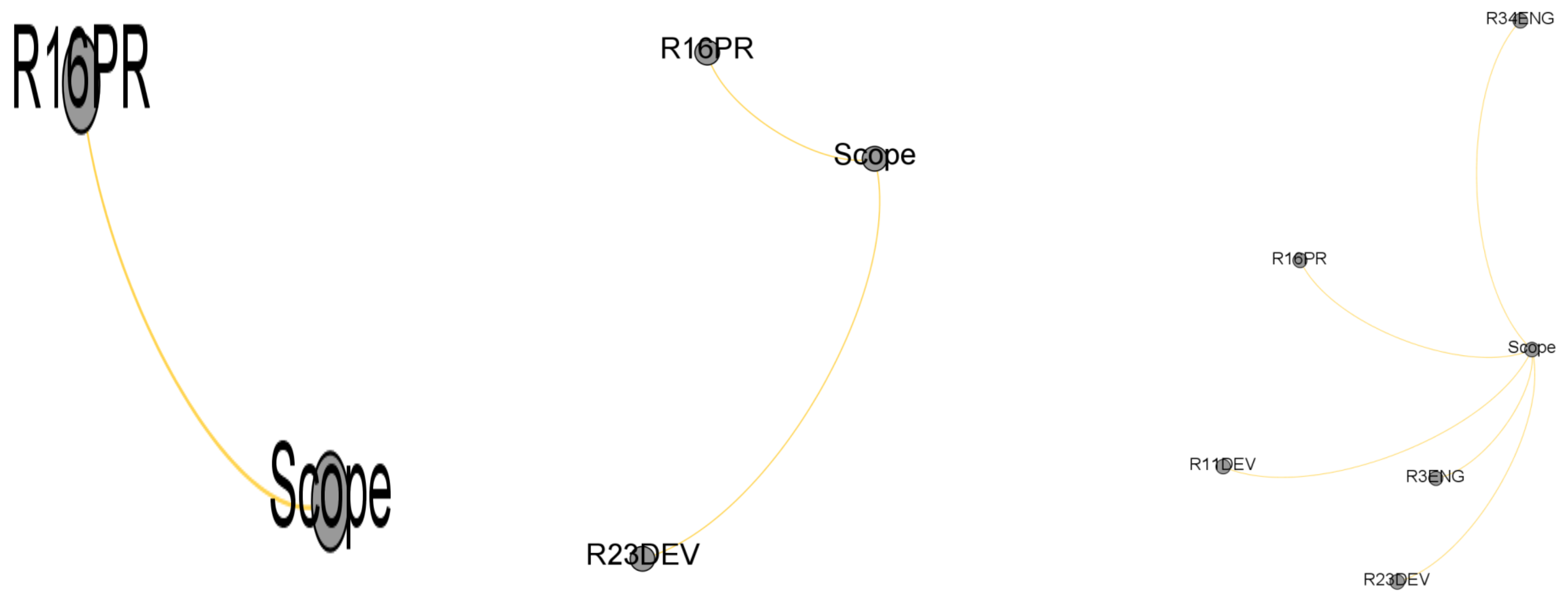


Figure 13-5S4DEV's influence on risk factors in the SCOPE ego network

It is noticeable that the most central node is S4DEV Effective project management, as it has a connections with 58.14% of the total number of nodes available in this network. S4DEV is also the most central factor in the network in terms of its betweenness value of 88.82 . Furthermore, S4DEV is connected to 48 risk factors connected to the SCOPE ego network which, in other words, indicates that effective project management has control and influence on all of the risk factors connected to the SCOPE ego network. It should also be noted that this factor is in close proximity to the rest of the risk factors; this is supported its closeness centrality value of 1.42.

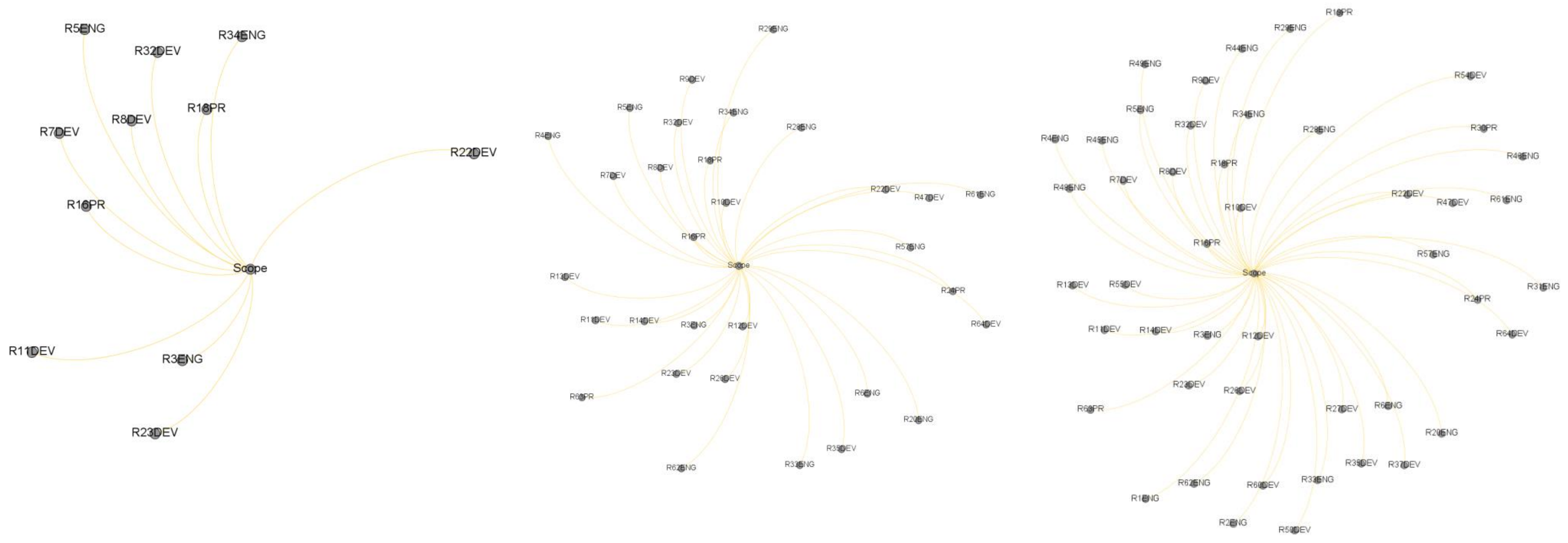


This graph shows the risk factors with a degree of 37 and more in the scope ego network

This graph shows the risk factors with a degree of 36 and more in the scope ego network

This graph shows the risk factors with a degree of 35 and more in the scope ego network

Figure 13-6 The change that occurs in the scope ego network and its risk factors when adjusting the degree topology value part one



This graph shows the risk factors with a degree of 34 and more in the scope ego network

This graph shows the risk factors with a degree of 30 and more in the scope ego network

This graph shows the risk factors with a degree of 25 and more in the scope ego network

Figure 13-7 The change that occurs in the scope ego network and its risk factors when adjusting the degree topology value part two

13.3.2 *Success factors influencing SCOPE*

As can be seen above in the graph relating to the degree range of 37, only one factor is connected to SCOPE, R16PR Unrealistic budget”. When reducing the minimum degree range to 36, the number of factors connected to SCOPE increases to two. In the minimum degree range of 35, there are five factors connected to the SCOPE ego network. Eleven factors are respectively connected to SCOPE when the degree of 34 is applied. As can be seen above, there are 31 factors with a degree of 30 or higher. The last degree range applied in the SCOPE ego network is in the minimum degree range of 25 with 47 factors interacting with SCOPE directly.

13.3.2.1 *The risk factor that has the most control and influence over the success factors in scope ego network*

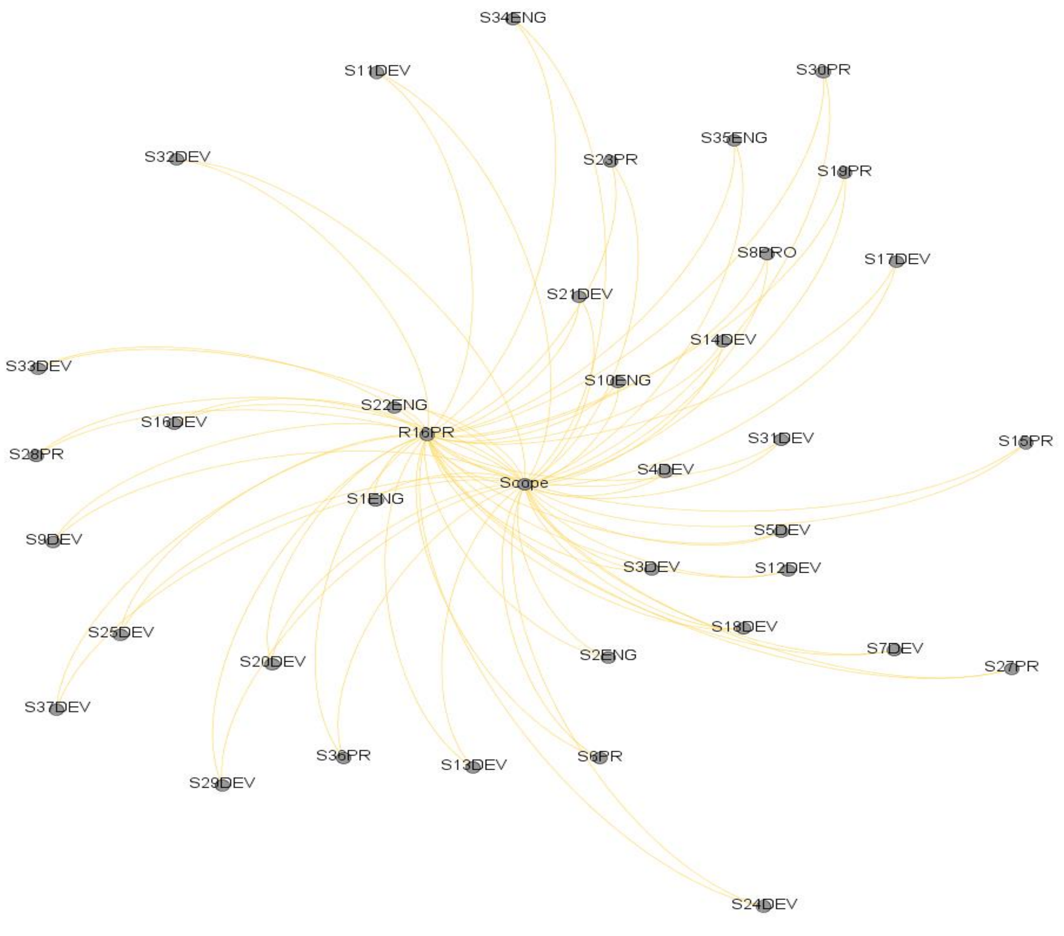


Figure 13-8R43PR’s influence on success factors in the SCOPE ego network

As mentioned above, the centrality measures will be used to determine the most central risk factor and find how it influences the SCOPE network as well as exploring the interactions in its isolated network. It is noticeable that the most central node is R16PR Unrealistic budget in terms of its degree, closeness and betweenness centrality, as has been mentioned previously. Furthermore, as R16PR is connected to 36 success factors connected in the scope ego risk network but is not connected to S26DEV Customer training and education or S38 Project managers dedicated to the project. It is also worth mentioning that R16PR is connected to 44% of the total number of nodes available in this network.

13.4 Modelling of the relationship between the isolated network nodes

Models

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to SCOPE	.998[a]	.996	.993	.04771	.996	309.312	49	57	.000
Success to SCOPE	1.000[a]	.999	.999	.01826	.999	3602.919	37	69	.000

13.4.1 *Risk factors association model results*

The Risk to SCOPE sub-model is associated with a large number of risk impacts. There is a 99% association between Risk to SCOPE and their risk impact events, which reflects the influence of S4DEV on the risk factors' interaction with success. The p-values shown above determine how good a fit the model is for the observed data. The p-values that are ≤ 5% mean that the null hypothesis can be rejected. As can be seen, the p-values in Risk to SCOPE are 000% ≤ 5%, which rejects the null hypothesis. This model is likely to show a real representation of the association between dependent and independent variables. The p-values in Success to scope shown above show that the null hypothesis can be rejected as the p-values is ≤ 5%. There is a 99% association between success to SCOPE and their risk impact events, which reflects the influence of R16PR on the risk factors' interaction with success.

Table 13-3 Risk factors association model results for Risk-to-scope

Model	Unstandardised Coefficients		Sig.	Model	Unstandardised Coefficients		Sig.
	B	Std. Error			B	Std. Error	
(Constant)	-.073	.055	.170	R29ENG	.007	.012	.584
R1ENG	.015	.010	.201	R30PR	.021	.010	.046
R2ENG	.006	.008	.635	R31ENG	.026	.010	.014
R3ENG	.017	.010	.178	R32DEV	.018	.010	.087
R4ENG	.024	.008	.010	R33ENG	-.005	.008	.501
R5ENG	.025	.010	.017	R34ENG	.018	.011	.123
R6ENG	.017	.008	.123	R35DEV	.006	.011	.576
R7DEV	.020	.010	.100	R37DEV	.032	.008	.000
R8DEV	.026	.008	.004	R43PR	.010	.013	.430
R9DEV	.016	.009	.166	R44ENG	.028	.012	.018
R10DEV	.013	.010	.130	R45ENG	.021	.011	.067
R11DEV	.034	.010	.004	R46ENG	.015	.011	.178
R12DEV	.018	.008	.047	R47DEV	.026	.009	.008
R13DEV	.004	.008	.713	R48ENG	.023	.009	.014
R14DEV	.025	.010	.031	R49ENG	.018	.009	.049
R16PR	.014	.011	.136	R50DEV	.035	.013	.012
R18PR	.014	.010	.086	R54DEV	.040	.010	.000
R19PR	.020	.010	.011	R55DEV	.019	.010	.055

R20ENG	.038	.007	.002	R57ENG	.011	.012	.350
R22DEV	.039	.006	.000	R60DEV	.029	.010	.005
R23DEV	.011	.009	.239	R61ENG	.025	.010	.019
R24PR	.033	.011	.001	R62ENG	.012	.011	.241
R26DEV	.018	.009	.047	R63PR	.035	.009	.001
R27DEV	.032	.011	.001	R64DEV	.023	.010	.021
R28ENG	.036	.006	.001	S4DEV	.016	.009	.067
	a. Dependent Variable: RISK-TO-SCOPE						

It noticeable that the R54DEV No update plan to the final software product has the largest beta coefficient of 0.04, which means this factor has the strongest unique contribution on project RISK-TO-SCOPE . For every unit increase in R54DEV a 0.04 unit increase in RISK-TO-SCOPE is predicted. Also, the Beta value of 0.039 for R22DEV Improper planning was slightly lower than R54DEV, indicating that it made less of a contribution to SCOPE. Those two high contribution factors show the importance of the development class, especially the planning stage. Two points has been noticed about this network and how the factors contribute to risk-to-scope . First, all the risk factors in this multiple regression have a positive impact on SCOPE except for R33ENG Size of the project. Second, it is also noticeable that most factors make a significantly low contribution to the rSCOPE where the strongest, as has been mentioned has a contribution value less than 0.04. This shows that, in relation to rscope, the is no dominant contribution factor; instead, a group of factors have a noticeable contribution, as six of the top 10 factors were from the Development class, as the total contribution is 0.499 compared to the Engineering class’s contribution of 0.383 or Program Constraints class’s contribution of 0.146. This shows the importance of the Development class’s impact on the scope of software project risk.

13.4.2 *Success factors association model results*

Table 13-4 Risk factors association model results for success-to-scope

Model	Unstandardised Coefficients			Model	Unstandardised Coefficients		
	B	Std. Error	Sig.		B	Std. Error	Sig.
(Constant)	-.004	.013	.733	S19PR	.024	.005	.000
S1ENG	.025	.004	.000	S20DEV	.034	.005	.000
S2ENG	.029	.004	.000	S21DEV	.020	.004	.000
S3DEV	.025	.004	.000	S22ENG	.021	.004	.000
S4DEV	.027	.004	.000	S23PR	.035	.004	.000
S5DEV	.028	.003	.000	S24DEV	.037	.004	.000
S6PR	.034	.003	.000	S25DEV	.024	.004	.000
S7DEV	.024	.004	.000	S27PR	.027	.003	.000
S8PR	.023	.003	.000	S28PR	.043	.004	.000
S9DEV	.024	.004	.000	S29DEV	.020	.004	.000
S10ENG	.029	.004	.000	S30PR	.036	.003	.000
S11DEV	.025	.004	.000	S31DEV	.026	.005	.000
S12DEV	.032	.004	.000	S32DEV	.025	.004	.000
S13DEV	.019	.004	.000	S33DEV	.020	.005	.000
S14DEV	.023	.004	.000	S34ENG	.026	.004	.000
S15PR	.026	.004	.000	S35ENG	.021	.004	.000
S16DEV	.029	.004	.000	S36PR	.030	.005	.000
S17DEV	.032	.003	.000	S37DEV	.054	.005	.000
S18DEV	.025	.003	.000	R16PR	-.001	.003	.859
	a. Dependent Variable: SUCCESS-TO-SCOPE						

From above, it seems that S37DEV Team training makes the biggest contribution to project SUCCESS-TO-SCOPE with a value of 0.054. This result shows the importance of improving the team's skills using continuous training and how it affects the project SCOPE, as an increase of unit of S37DEV has a positive impact on project SUCCESS-TO-SCOPE with an increase in value of 0.054. It is interesting to note that S28PR Good performance by vendors/contractors/consultants has relatively the second strongest contribution to SCOPE with a beta value of .043, as it shows the importance of having a good relationship with and contribution by vendors to the project scope.

13.5 The association between the network nodes

Table 13-5 S4DEV and Risk-to-scope association network

Factors	RISK-TO-SCOPE			S4DEV		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
R1ENG	.130	.181		.030	.759	R1ENG
R2ENG	.399**	.000		.126	.197	R2ENG
R3ENG	.506**	.000		.274**	.004	R3ENG
R4ENG	.520**	.000		.242*	.012	R4ENG
R5ENG	.553**	.000		.301**	.002	R5ENG
R6ENG	.239*	.013		.047	.630	R6ENG
R7DEV	.626**	.000		.368**	.000	R7DEV
R8DEV	.519**	.000		.091	.351	R8DEV
R9DEV	.598**	.000		.133	.173	R9DEV
R10DEV	.550**	.000		.261**	.007	R10DEV

R11DEV	.624**	.000		.272**	.005	R11DEV
R12DEV	.482**	.000		.356**	.000	R12DEV
R13DEV	.639**	.000		.293**	.002	R13DEV
R14DEV	.640**	.000		.484**	.000	R14DEV
R16PR	.646**	.000		.186	.055	R16PR
R18PR	.570**	.000		.131	.179	R18PR
R19PR	.525**	.000		.095	.329	R19PR
R20ENG	.536**	.000		.251**	.009	R20ENG
R22DEV	.505**	.000		.237*	.014	R22DEV
R23DEV	.452**	.000		.276**	.004	R23DEV
R24PR	.614**	.000		.093	.341	R24PR
R26DEV	.641**	.000		.288**	.003	R26DEV
R27DEV	.708**	.000		.354**	.000	R27DEV
R28ENG	.585**	.000		.206*	.034	R28ENG
R29ENG	.716**	.000		.341**	.000	R29ENG
R30PR	.628**	.000		.168	.083	R30PR
R31ENG	.769**	.000		.343**	.000	R31ENG
R32DEV	.581**	.000		.218*	.024	R32DEV
R33ENG	.577**	.000		.240*	.013	R33ENG
R34ENG	.702**	.000		.513**	.000	R34ENG
R35DEV	.537**	.000		.379**	.000	R35DEV
R37DEV	.642**	.000		.255**	.008	R37DEV
R43PR	.726**	.000		.354**	.000	R43PR
R44ENG	.680**	.000		.357**	.000	R44ENG

R45ENG	.743**	.000		.369**	.000	R45ENG
R46ENG	.698**	.000		.284**	.003	R46ENG
R47DEV	.540**	.000		.252**	.009	R47DEV
R48ENG	.717**	.000		.253**	.009	R48ENG
R49ENG	.735**	.000		.382**	.000	R49ENG
R50DEV	.719**	.000		.375**	.000	R50DEV
R54DEV	.643**	.000		.268**	.005	R54DEV
R55DEV	.666**	.000		.317**	.001	R55DEV
R57ENG	.793**	.000		.364**	.000	R57ENG
R60DEV	.690**	.000		.192*	.047	R60DEV
R61ENG	.761**	.000		.373**	.000	R61ENG
R62ENG	.326**	.001		.195*	.044	R62ENG
R63PR	.723**	.000		.332**	.000	R63PR
R64DEV	.609**	.000		.177	.068	R64DEV
Risk-to-scope	1			.452**	.000	Risk-to-scope
S4DEV	.452**	.000		1		S4DEV

13.5.1 *Risk to SCOPE*

The results show that 38 factors are significantly correlated with S4DEV Effective project management. Thirty-one factors have a correlation that is significant at the 0.01 level and seven factors have a correlation that is significant at the 0.05 level. R34ENG Incompatible development environment has the strongest and most positive relationship with S4DEV of all the risk factors [r= 0. 513, p<.0005]. Furthermore, R14DEV Inefficient team capability has the second strongest and most positive correlation with S4DEV [r= 0. 484, p<.0005]. This research also noted that all factors have a positive correlation with S4DEV. The results also show that there was a strong, positive correlation between the risk SUCCESS-TO-SCOPE and R57ENG Use of cheap tools [r= 0. 793, p<.0005], with high levels of use of cheap tools correlating with high levels of project risk-to-scope , which makes it the strongest relationship factor with risk-to-scope . It is worth mentioning that R57ENG has also been recognised by this research to have to strongest relationship as well with risk-to-time , which stresses the importance of this factor in interaction with those criteria.

R31ENG Improper marketing techniques also has a positive and strong relationship with project RISK-TO-SCOPE [r= 0. 769, p<.0005]. It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised as having a correlation that is significant at the 0.01 level with the risk-to-SCOPE; the only factor that does not have a significant correlation with risk-to-scope is R6ENG Requirement creep, as the correlation with risk-to-scope was significant at the 0.05 level. Furthermore, R1ENG Unclear customer requirements has not been recognised by these results to have a significant correlation with risk-to-scope .

13.5.2 *Success factors influencing SCOPE*

Table 13-6 R16PR and Success-to-scope association network

Factors	SUCCESS-TO-SCOPE			R16PR		Factors
	Pearson Correlation	Sig. (2-tailed)		Pearson Correlation	Sig. (2-tailed)	
S1ENG	.391**	.000		.143	.142	S1ENG
S2ENG	.546**	.000		.181	.062	S2ENG
S3DEV	.724**	.000		.326**	.001	S3DEV
S4DEV	.734**	.000		.186	.055	S4DEV
S5DEV	.603**	.000		.242*	.012	S5DEV
S6PR	.634**	.000		.318**	.001	S6PR
S7DEV	.763**	.000		.347**	.000	S7DEV
S8PR	.679**	.000		.501**	.000	S8PR
S9DEV	.745**	.000		.322**	.001	S9DEV
S10ENG	.746**	.000		.244*	.011	S10ENG
S11DEV	.732**	.000		.206*	.034	S11DEV
S12DEV	.645**	.000		.191*	.049	S12DEV
S13DEV	.652**	.000		.234*	.015	S13DEV
S14DEV	.756**	.000		.421**	.000	S14DEV
S15PR	.709**	.000		.369**	.000	S15PR
S16DEV	.830**	.000		.429**	.000	S16DEV
S17DEV	.660**	.000		.214*	.027	S17DEV
S18DEV	.604**	.000		.213*	.028	S18DEV

S19PR	.793**	.000		.313**	.001	S19PR
S20DEV	.772**	.000		.321**	.001	S20DEV
S21DEV	.751**	.000		.265**	.006	S21DEV
S22ENG	.699**	.000		.225*	.020	S22ENG
S23PR	.721**	.000		.121	.214	S23PR
S24DEV	.741**	.000		.239*	.013	S24DEV
S25DEV	.784**	.000		.317**	.001	S25DEV
S27PR	.671**	.000		.269**	.005	S27PR
S28PR	.778**	.000		.369**	.000	S28PR
S29DEV	.754**	.000		.306**	.001	S29DEV
S30PR	.589**	.000		.274**	.004	S30PR
S31DEV	.770**	.000		.309**	.001	S31DEV
S32DEV	.675**	.000		.381**	.000	S32DEV
S33DEV	.786**	.000		.367**	.000	S33DEV
S34ENG	.794**	.000		.408**	.000	S34ENG
S35ENG	.816**	.000		.324**	.001	S35ENG
S36PR	.707**	.000		.287**	.003	S36PR
S37DEV	.797**	.000		.456**	.000	S37DEV
Success-to-scope	1			.420**	.000	Success-to-scope
R16PR	.420**	.000		1		R16PR

The results show that 33 factors are significantly correlated with R16PR. Twenty-four factors have a correlation that is significant at the 0.01 level and nine factors have a correlation that is significant at the 0.05 level. There was a strong, positive correlation between R16PR Unrealistic budget and S8PR Realistic budget [r= 0. 501, p<.0005], which again shows the importance of this factor in relation to the project budget. Also, it is noted that S8PR has a stronger and more positive relationship with SUCCESS-TO-SCOPE [r= 0. 679, p<.0005]. S37DEV Team training has the second strongest and most positive relationship with success to R16PR [r= 0. 456, p<.0005] as a trained team will usually have a good impact on the project budget. The results also show that there was a strong, positive correlation between SUCCESS-TO-SCOPE and S16DEV Good leadership [r= 0. 830, p<.0005]. It is understandable that S35ENG Extensive testing for quality has the second strongest relationship with SUCCESS-TO-SCOPE [r= 0. 816, p<.0005], which shows the importance and the link between quality and scope. It is worth mentioning that all the correlations in this network have a positive relationship where all the factors have been recognised as having a correlation that is significant at the 0.01 level with the SUCCESS-TO-SCOPE , which agrees with this research's claim of their importance to what have been mentioned as important to the SCOPE ego network as well as their interaction with R16PR. Although R16PR has one of lowest relationship strengths, that could be because, as has been mentioned, the network was about the influence of the most central risk factor where all the factors have been correlated from a success point of view to success-to-SCOPE, which means R16PR has a huge but indirect relationship with success-to-SCOPE.

13.6 Summary

This chapter has investigated the interdependency of the scope ego network, and analysed the ego network general characteristics. Furthermore, the factors with R16PR and S4DEV have been found to be the most controlling factors in this network. The associations between the factors in relation to controlling factors have also been analysed and investigated.

Chapter 14: Discussion

Chapter 14: Discussion

14.1 Introduction

This chapter presents a discussion of the main research themes reported throughout this thesis. The first section presents a discussion of the research findings on the descriptive results, whilst the second section discusses the results from the fuzzy mapping, and the third section discusses the results from the interaction network.

14.2 Research question 1

What are the most important risk factors and their classification in software development projects?

14.2.1 *Product Engineering class*

The Product Engineering class covers the technical aspects of software development, and has five sub-classes (requirements, design, code, unit testing integration & test, and engineering specialties). This research has identified 25 risk factors related to this class, six of which were ranked in the first 20 highest indicators, namely: R1ENG Unclear customer requirements, R2ENG Unable to meet user requirements, R3ENG Lack of technical skills, R20ENG Understanding problems of customers, R29ENG Inappropriate technology and R62ENG Unclear or misunderstood scope/objectives. R1ENG Unclear customer requirements has ranked as the most important factor in this class as well as in all the other classes in software development. This result agrees with Sonchan and Ramingwong (2014), about the top 20 risk factors in software development: they found that unclear customer requirements is the most-cited factor in research papers on software risk management. It usually occurs when customers (users') are finding it difficult to clearly identify the requirements needed from the software before the project starts. Although from a risk exposure point of view they found requirement creep as the most important risk factor, they concluded that unclear customer requirements is one of the most challenging risks. This was later emphasised by their 2015 research on simulating risk management in software engineering, where they found that unclear customer requirements was ranked as the factor that occurs most often during a software development project. Also, Oteyo et al. (2014) observed that: “Failure of software projects is dependent on unclear customer requirements, lack of development strategy and the outcome is transfer of blame to the human resources behind the development process”. In addition, Philge (2014) noted that unclear customer requirements lead to developers and test team finding it difficult to test the software due to ambiguous customer expectations of the system. R20ENG Understanding problems of customer has ranked as the second most important factor. According to Buse and Zimmermann (2012), software projects are not excepted from the importance of understanding the problems of customers, as this is important to all types of business projects. They emphasise the channels that should be used to make the customers’ problems clearer, like crash reports, telemetry and surveys. This research found that the development team ranked this factor as the second most important factor in software projects, whilst the management team ranked it as the seventh most important factor. This could be because misunderstanding of the customers’ problems usually occurs in the development team, as Al-Karaghouli et al. (1999) argued: the failure of a significant number of information technologies and systems is due to the misunderstanding that occurs between the software engineers and the customers. They claim that the software engineers do not always understand the customers’ problems and business needs, but often the customers do not take into consideration the reality of the limited ability of software outcome and the development team can offer. Also, Buse and Zimmermann (2012) noted that “Making customer data actionable implies directly relating to development effort. Not only must we know which features are valuable or problematic, it must also be possible to identify these features in the source code, to track their progress, and to employ customer feedback to guide specific aspects of development and maintenance”. Usually, all of those steps lie in the hands of the development team. The main reason why this factor ranked so high with the development team is due to customers’ lack of understating of the developers’ technical terminologies (Asif et al., 2014).

Furthermore, this study’s results show that the respondents found that factors R2ENG Unable to meet user requirements, R3ENG Lack of technical skills, R62ENG Unclear or misunderstood scope/objectives and R29ENG Inappropriate technology also make an important contribution to product engineering as they were ranked in the top 20 factors. It is noticeable that there are differences in ranking the importance of each factor in this class from the perspectives of the management and development teams, the development team has ranked R3ENG Lack of technical skills as 21st in terms of its importance, compared with the management team, who have ranked it as 9th. However, in terms of a statistically significant difference between the two groups, the t-test showed that the only factor that has a significant different in this class is R28ENG Inappropriate design [R28ENG = .016, $p < .05$], as is shown in more detail in below:

Table 14-1 Product Engineering and hypothesis test

Research question	***Product Engineering class***
Hypothesis	H_0 *(p>0.05)*
Results	The t-test results indicated that: there were significant differences between the survey participants regarding risk factor: R28ENG: "inappropriate design"
Researcher's observation	Inappropriate design was ranked as 15th in the overall ranking by the development team whilst the management team ranked it as 41st. This result shows that the development team are more aware of the importance of the design to the success or failure of a project, while software designers in the development team understand the importance of this factor as, according to Asif et al. (2014), "Software designers have a major role in the success or failure of the project". The reason why managers gave inappropriate design a low score could be due to their focus on the services provided rather than on the design itself, as the KSA market for software products is a relatively new market and the competition is about providing the service without paying much attention to the design.
Conclusion	The null hypothesis was rejected for R28ENG: Inappropriate design. The null hypothesis (H0: 1=0) - (p > 0.05) was retained for other product engineering factors.

14.2.2 *Development Environment class*

The Development Environment class contains risks related to the project environment, like methods, procedures and tools used to develop the software (Menezes Jr et al., 2013). It contains five elements: Development System, Management Process, Development Process, Management Methods and Work Environment (Kibe, 2015). This class consists of 26 risk factors, and the research found that it dominated the top 20 risk factors, with nine out of the top 20 – more than any other class. The nine factors (R22DEV Improper planning, R32DE

Lack of top management commitment and support, R14DEV Inefficient team capability, R23DEV Lack of experience of project manager, R35DEV Unavailable customer contact, R47DEV Design is skipped or is created after code is written, R9DEV Inadequate infrastructure, R10DEV Unrealistic schedule and R12DEV Communication gaps) have been recognised by the participants as very important to a software project as they have a mean score between 4.15 and 4.32. Factor R22DEV Improper planning has been ranked as the most important factor in this class as it has a mean value of 4.32, and it is also ranked as the 3rd most important factor in software project development. According to Hu et al. (2013), the importance of risk planning is to “make sure that the consequences and the sources of the risk are known”. Furthermore, good software project management is usually due to the implementation of good risk planning (Hu et al., 2013). This research result agrees with Aloini et al. (2007), who mentioned that improper planning played a major role in the frailest of 40% of software projects reviewed in their study. It is interesting to find that Demir (2009b) found that the participants in his research ranked improper planning as (Demir, 2009a)the third most important challenge faced in managing a software project – which is the same ranking as this study has found. Furthermore, his study found that requirement management was one of the top challenges, but it is also worth mentioning that R20ENG Understanding problems of customers has not been included as an important factor in his study. One of the most important implications of poor planning is its effect on time as well as the spiralling cost to the project. The main risk caused by this factor is due to the fact that a software project team gives more attention to other aspects which are less important than planning where, for example, according to Serrador and Turner (2015) about 43% of the focus has been paid to software testing whilst planning and requirements have received 6%. They noted that if more attention was paid to planning it would help in cost saving . Serrador (2014) argued in his book that some mangers think that “It is better to skip the planning and to start developing the requested system”. This research, after looking at the literature review in this regard, agrees with what Serrador (2014) claimed that some managers' reasons for skipping the planning phase are not valid.

This research has found that there is no significant difference between the management and development teams with regard to their opinion on the importance of this factor as the management team have ranked it as 3rd and the development team have ranked it as 6th. Apart from factor R22DEV, three more factors, R32DE, R14DEV, and R23DEV, were ranked in the top 10 as they have an overall ranking of 5th, 7th and 8th (out of 64) respectively. R32DEV Lack of top management commitment and support has been ranked by the participants as the second most important factor in this class as it has a mean of 4.31. This factor's importance is really based on the fact that if the top management are not committed and supported to a project this will create a general environment of uncertainty for the project, which could affect the project's needs and scope. For example, if the top management were not committed to a project, it could be less of a priority for them and they may thus not invest in new technology for it due to the cost, which they might believe might not be recouped (Liu et al., 2010). Moreover, Liu et al. (2010) found that this factor was ranked as the most important factor by their Chinese participants. Schmidt et al. (2001b)have also ranked this factor as the most important factor, but more recent studies (Altahtooh and Emsley, 2014b, Lehtinen et al., 2014, Elzamly and Hussin, 2014) have ranked this factor as one of the top 10 factors in their respective research. As a result of that, this research believes that this factor is an important contributor to the success or failure of a software project. This research has found that there are no significant differences between the management and development teams with regard to their opinion on the importance of this factor as the management team have ranked it as 6th with a mean of 4.42 and the development have ranked it as 4th with a mean of 4.17. The only factor that has a significant difference, as the T-test has shown, is R23DEV Lack of experience of project managers, as is shown in more detail in Table 14-2.

Table 14-2 Development Environment and hypothesis test

Research question	***Development Environment class***
Hypothesis	*H_0 (p>0.05):*
Results	The t-test results indicated that: there were significant differences between the survey participants regarding risk factor: R23DEV Project manager lacks experience
Researcher's observation	Two points have been noticed with regard to this factor. First, that this factor is one of the top 20 risk factors as it ranked as the 8th most important factor in the overall ranking and 4th in this class, which shows its importance. Second, the development team ranked it as the 16th most important factor in a software project; in contrast, the management team ranked it as the 2nd most important factor. Managers usually depend on experience when making a decision as, although there is a lot of information related to a software project, there is a difference between the information needed and the information available and, according to Buse and Zimmermann (2012), mangers tend to rely on their experience when the necessary information is not available due to many reasons (e.g., tools are unavailable, too difficult to use, too difficult to interpret, or they simply do not present useful or actionable information). Also, managers are more aware of the importance of this risk due to their understanding of the consequences of project failure due to an incorrect decision being made by inexperienced managers (Pressman, 2010, Westfall, 2000) . Although developers understand the importance of having an experienced project manager, the management team are more alert to the implications that this factor could have for a project.
Conclusion	The null hypothesis was rejected for R23ENG Lack of experience of project managers. The null hypothesis (H_1: 1=0) - (p > 0.05) was retained for the other development environment factors.

14.2.3 *Program Constraints*

The Program Constraints class covers all the factors that have an effect on a project but which cannot be controlled by the project or, in other words, it covers all the factors that are external to the project (Uzzafer, 2013, Hill, 2007). This class has three sub-classes (Resources, Contract and Program Interfaces). This class consists of 13 risk factors. These factors' means range from 3.56 to 4.31. The score of the average weighted mean for all of these indicators is 3.95 and is very high in comparison with the other classes. Five factors from the 13 factors in this class were ranked in the 20 highest ranked indicators, with three of these in the top 10 highest ranked indicators. They are R41PR Lack of staff experience, R16PR Unrealistic budget and R18PR User resistance. Lack of staff experience has an overall rank of 4th. The management team ranked it as the 4th most important factor, whilst the development team ranked it as the 8th most important factor to a software project. This result shows that both teams are aware of the importance of this factor as they have both ranked it in the top 10 factors. Many aspects can be affected by lack of experience, where the method and frequency of communication among a project's staff is affected by the level of experience that they have. Cost estimation can be greatly affected by staff experience, as some tasks could be performed easily and quickly if they have been done before in a similar project. Staff experience is also considered the most important factor during the cost estimation (Potda et al.). According to Sarigiannidis and Chatzoglou (2014), staff experience is one of the most important factors that must be taken into consideration when measuring the quality of a software project's staff. The complexity of execution of the project could be disturbed and affected by the staff's lack of experience (Miorando et al., 2014). Also, Gandhi et al. (2014) noted that lack of staff experience has been considered over the years to be one of the main reasons for a project's failure or success.

R16PR Unrealistic budget has an overall rank of 6th with a mean of 4.30. The management team ranked this factor in the top 10 (as 5th) and the development team ranked it as 11th. The result shows that both teams are aware of this factor's importance. The factor has been mentioned by Boehm (1991) as one of the top 10 risk factors affecting software development projects. Also, Sipayung and Sembiring (2015), Sonchan and Ramingwong (2014) identified it as one the top 20 risk factors that have an impact on software projects. According to Shahzad and Iqbal (2007), required time, effort and resources are the main elements on which a budget is built. Furthermore, this factor usually becomes important and noticeable at the end of a project (Hoermann et al., 2011), where the first stages of the project will consume the budget if the cost of the project has been underestimated. The challenge behind this factor is because it can also be used a success criterion, where the project can be measured as a successful project, according to Hashimi et al. (2012), if it meets the estimated budget.

R18PR User resistance has an overall rank of 10th with a mean of 4.2. Kim and Kankanhalli (2009) define user resistance as relating to users who are targeted in a software project but who are not eager to change or to use the new or updated software. Their resistance could happen at any stage of the software project. This factor could be important for several reasons, according to Vrhovec et al. (2015), Lapointe and Rivard (2005): "inaction, distance, lack of interest, delay tactics, excuses, persistence of former behaviour, withdrawal, voicing opposite points of view, asking others to intervene or forming coalitions in the most aggressive manifestations, user resistance seeks to be disruptive and may even be destructive, e.g., infighting, making threats, strikes, boycotts and sabotage".

This research has not found any statistically significant differences in this class between the management and development teams, as the t-test has shown in more detail in Table 14-3.

Table 14-3 Program Constraints and hypothesis test

Research question	***Program Constraints class***:
Hypothesis	*H_0 (p>0.05):*
Results	The t-test results indicated that: there were no significant differences between the survey participants regarding risk factors.
Researcher's observation	Two points have been noticed with regard to those factors. First, there are no huge differences in ranking between the two groups in the top 20; the only factor that has a noticeable difference is R18PR User resistance but it is not statistically significantly different. Second, although the development team ranked R18PR User resistance as the 3rd most important factor in this class whilst the management team ranked it as the 19th most important factor, it is noticeable that both teams are aware of the importance of this factor as they have both ranked it in the top 20. The reason behind the development team giving this factor this rank could be because it is their duty to identify if there is any resistance by the users during the project so they can use it to their advantage in managing the change (Yu et al., 2015). Although developers understand the importance of having an experienced project manager, the management team are more alert to the implications that this factor could have on a project. It is also interesting to find there is a noticeable connection between R18PR User resistance and R32DEV Lack of top management commitment as, according to Vrhovec et al. (2015), lack of management commitment is considered one of the main sources of user resistance as management should pay more attention and support to change management.
Conclusion	The null hypothesis (H_0: 1=0) - *(p>0.05)* was retained for all Program Constraints factors.

14.3 Research question 2

•What are the interdependencies (based on construct correlation and dependency matrix) between the research constructs?

Section 1: construct correlation

14.3.1 *The interdependence between research constructs from the construct correlation*

14.3.1.1 *Introduction*

In this section, this research identifies the top central factors in three different fuzzy networks in order to meet the objective of exploring the centrality and effect of each factor. Their centrality is investigated using four different measures (degree, closeness, betweenness and eigenvector).

14.3.1.2 *Statistical comparison of top five risk factors influencing success factors from fuzzy network*

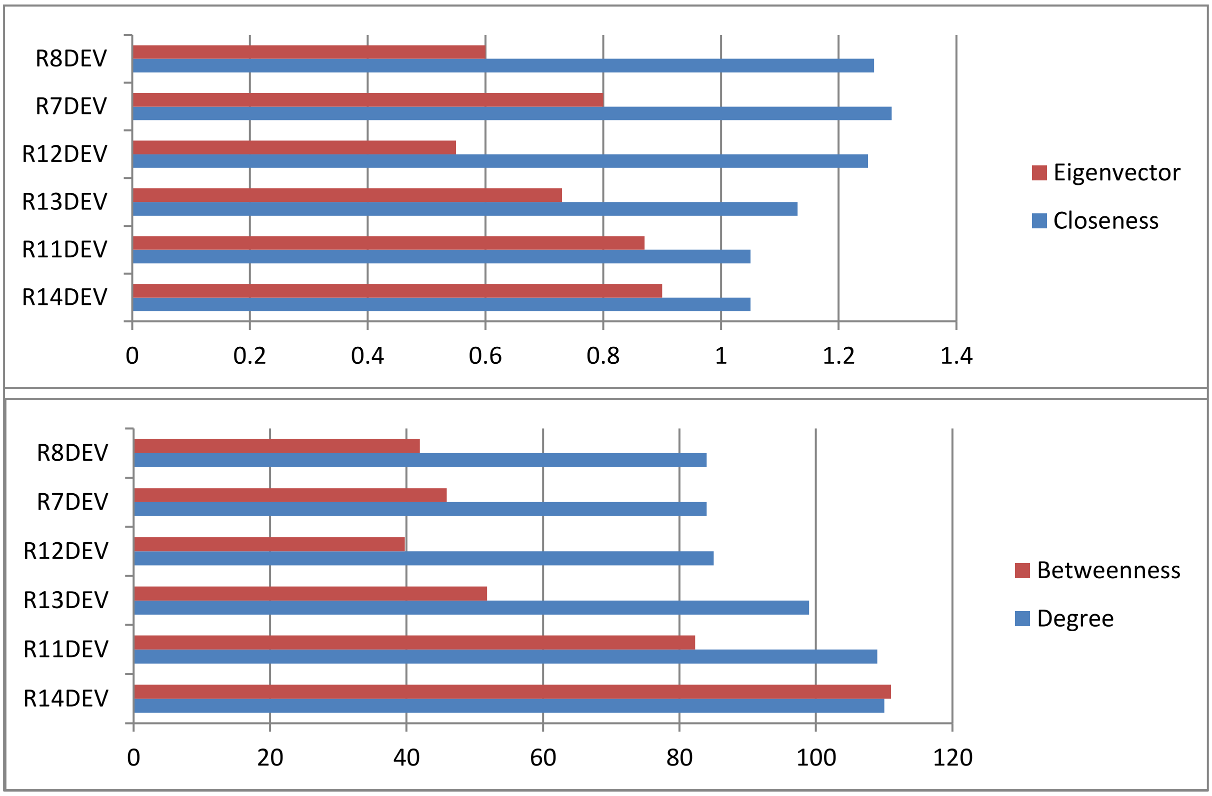


Figure 14-1 Most central factors' analysis for risk factors influencing success factors

From above, R14DEV Inefficient team capability is the most central factor in this network in terms of its direct impact on the other factors (degree) also, R14DEV is the quickest and most efficient to reach other risk and success factors in the network (Closeness). Furthermore, it has the most ability to control the interaction between the factors in this network (betweenness). This result agrees with Sipayung and Sembiring (2015), who found in their research about mapping the risk factors in software projects that the category that contained the inefficient team capability was the category with the most connections with other risk factors.

Table 14-4 Researcher's observation and Risk factors related to success factors

Network name	Risk factors influencing success factors
Sub-research question	*Do some factors have more centrality than others in this network?*
Results	This research found that the most central factors in the risk factors influencing success factors are: • R11DEV Unrealistic resource planning • R13DEV Conflicts among team members • R12DEV Communication gaps • R7DEV Inappropriate development process/methodology • R8DEV Problems with new technology
Researcher's observation	This research has found that, according to the centrality measures in degree, closeness and betweenness, R11DEV and R13DEV are the second and third top scores in all three measures. R12DEV Communication gaps has the sixth highest betweenness value of 39.76 but it is considered as the network's third most central factor as it has the third highest interrelationship with a connection to 85 factors and it has the third least shortest distance to all factors in the network. Furthermore, even though R7DEV Inappropriate development process/ methodology and R8DEVProblems with new technology each have an interrelationship with 84 factors, R7DEV is more central and has more impact on the network than R7DEV because it has more control over the other factors' connections to the network with a betweenness value of 45.9 compared to R8DEV's betweenness value of 41.97, although it has more shortest paths than R7DEV, but R7DEV has more connections to the top central factors with an eigenvector value of 0.8.
Conclusion	This result supports the claim that factors in this network are interrelated and some factors have more centrality than others in this network.

14.3.1.3 *Fuzzy networks of risk factors influencing success criteria*

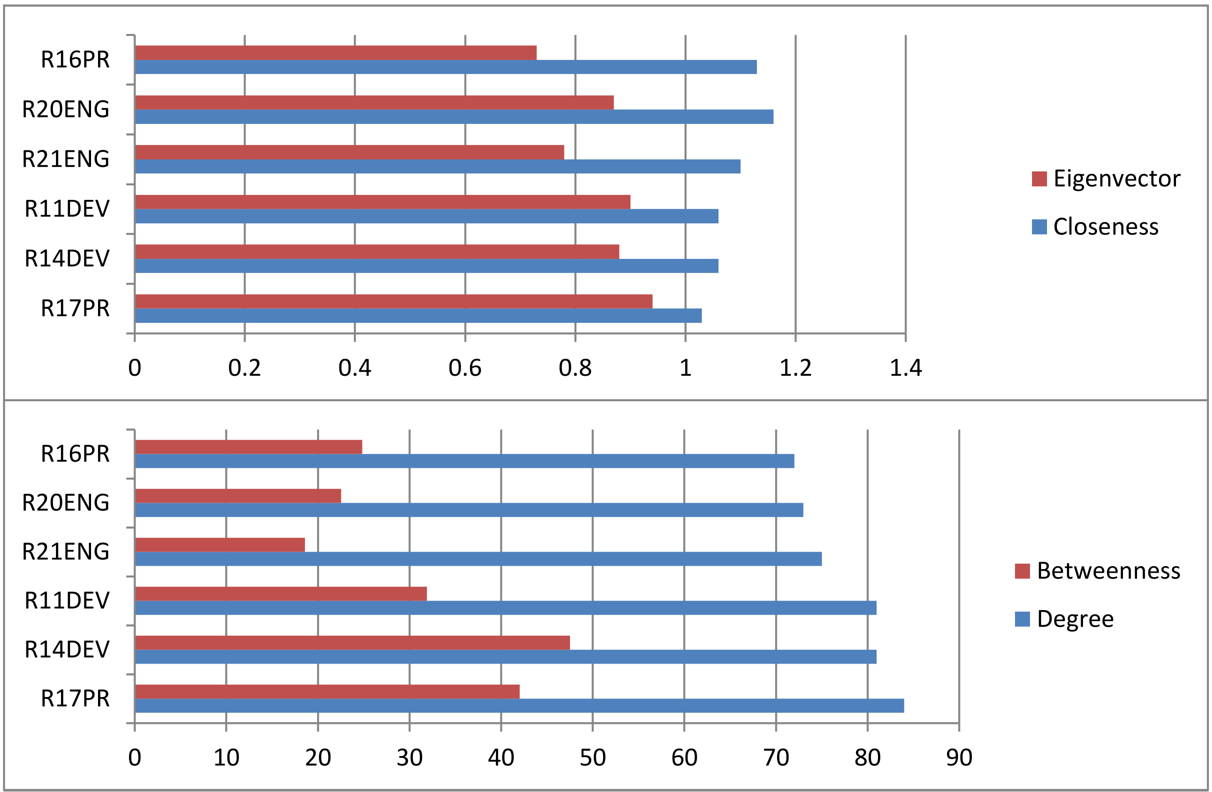


Figure 14-2 Most central factors' analysis for risk factors influencing success criteria

R17PR Resource insufficiency has been found to be the most central factor in risk factors influencing success criteria . It has the most direct connections to other factors is the controlling factor with the smallest distance to all other factors in this network. Its importance is based on its impact on other factors like "Time, assets, income, property and people" (Vosough, 2012). On the other hand, this research notes that Sonchan and Ramingwong (2014), in their research focusing on the top 20 risk factors, found resource insufficiency to be one of the factors that had less impact on a software project. This could be due to two reasons. First, it could be because network is a sub-network of impact fuzzy . Second, this research has not focused on the probability of occurrence; it has focused more on the centrality and important of this factor on success criteria and risk factors.

The results of this research agree with the 2015 CHAOS report, which found that a lack of resources is one of the top three factor that cause software development to be cancelled (Hastie and Wojewoda, 2015a, Safa'a, 2012).

Table 14-5 Researcher's observation and Risk factors related to success criteria

Network name	Risk factors influencing success criteria
Sub-question	*Do some factors have more centrality than others in this network?*
Results	This research found that the most central factors in the risk factors influencing success criteria are: • R14DEV Inefficient team capability • R11DEV Unrealistic resource planning • R21ENG Understanding problems of developers • R20ENG Understanding problems of customers • R16PR Unrealistic budget
Researcher's observation	This research has identified R14DEV, E11DEV and R21ENG as the network's second, third and fourth central factors according to their centrality measures scores. Although R16PR Unrealistic budget has the sixth highest number of connections with 72 connections to other factors and criteria in this network, however,because it has close distance to all other factors with a closeness value of 1.13 and has more shortest paths through it with a betweenness value of 24.85 than R20ENG Understanding problems of customers, which has been identified as the fifth most central factor whilst R20ENG Understanding problems of customers has ranked as the sixth.
Conclusion	This result supports the claim that factors in this network are interrelated and some factors have more centrality than others in this network.

14.3.1.4 *Fuzzy networks of success factors influencing success criteria*

In the success to criteria from impact fuzzy network, there are 14 factors with the same centrality of value to all other factors in the network, as is seen in the graph above. This result shows again that this network is well connected. Also, this research believes that the reason behind the noticeable number of central factors in addition to what has been mentioned is because of the criticality of the factors' high impact on the success of a project. This research has found that 12 of the most central factor have been recognised by Nasir and Sahibuddin (2011a) as the most critical factors in a software project. Furthermore, this network has the lowest number of factors compared to the other networks. This research has combined the interaction and impact to provide a more evidence of the significant differences relating to the success factors' impact on success criteria in the next chapter.

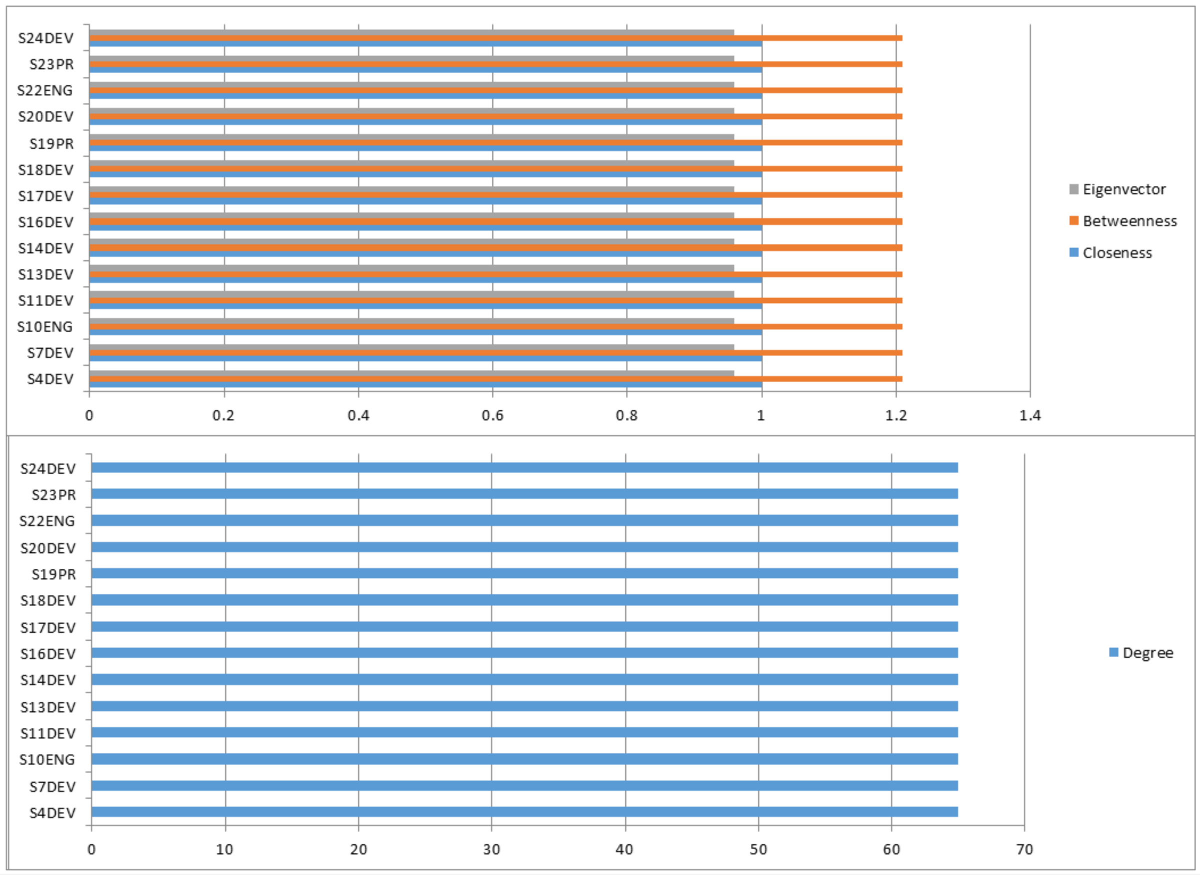


Figure 14-3 Most central factors' analysis for success factors influencing success criteria

In the success to criteria from impact fuzzy network, there are 14 factors with the same centrality of value to all other factors in the network, as is seen in the graph above. This result shows again that this network is well connected. Also, this research believes that the reason behind the noticeable number of central factors in addition to what has been mentioned is because of the criticality of the factors' high impact on the success of a project. This research has found that 12 of the most central factor have been recognised by Nasir and Sahibuddin

(2011a) as the most critical factors in a software project. Furthermore, this network has the lowest number of factors compared to the other networks. This research has combined the interaction and impact to provide a more evidence of the significant differences relating to the success factors' impact on success criteria in the next section, which discusses the third research question .

Table 14-6 Researcher's observation and Risk factors related to success criteria

Network name	Success factors influencing success criteria
Sub- question	*Do some factors have more centrality than others in this network?*
Results	This research found that the most central factors in the success factors influencing success criteria are: • S4DEV Efficient project management • S7DEV Effective communication and feedback • S10ENG Familiarity with technology/development methodology • S11DEV Appropriate development processes/methodologies • S13DEV Up-to-date progress reporting • S14DEV Effective monitoring and control • S16DEV Good leadership • S17DEV Risk management • S18DEV Change management • S19PR Appropriate infrastructure • S20DEV Committed and motivated team • S22ENG Complexity, project size, number of organisations involved • S23PR Pilot project performance • S24DEV Clear assignment of roles and responsibilities
Researcher's observation	This research has noticed that the betweenness value in this network is very low as the number shortest paths is low: 17 of the most controlled factors have a betweenness value between 1.02 and 1.21, which shows that the network is close to being a star network. It also reflects that it is difficult to control the network's impact due to the large number of central factors. Furthermore, this research noted that, although these factors have the same centrality value on the success factors in influencing related to success criteria network, the participants have ranked them differently. In addition, the average mean score given to the success factors by the participants is 4.25, which is also evidence of their importance, and also explains the high number of central factors.
Conclusion	This result supports the claim that factors in this network are interrelated and some factors have more centrality than others in this network.

Section 2: Network of interaction

14.3.2 *The interdependence between research constructs from the dependency matrix*

14.3.2.1 *The top central factors to all criteria in the interaction network*

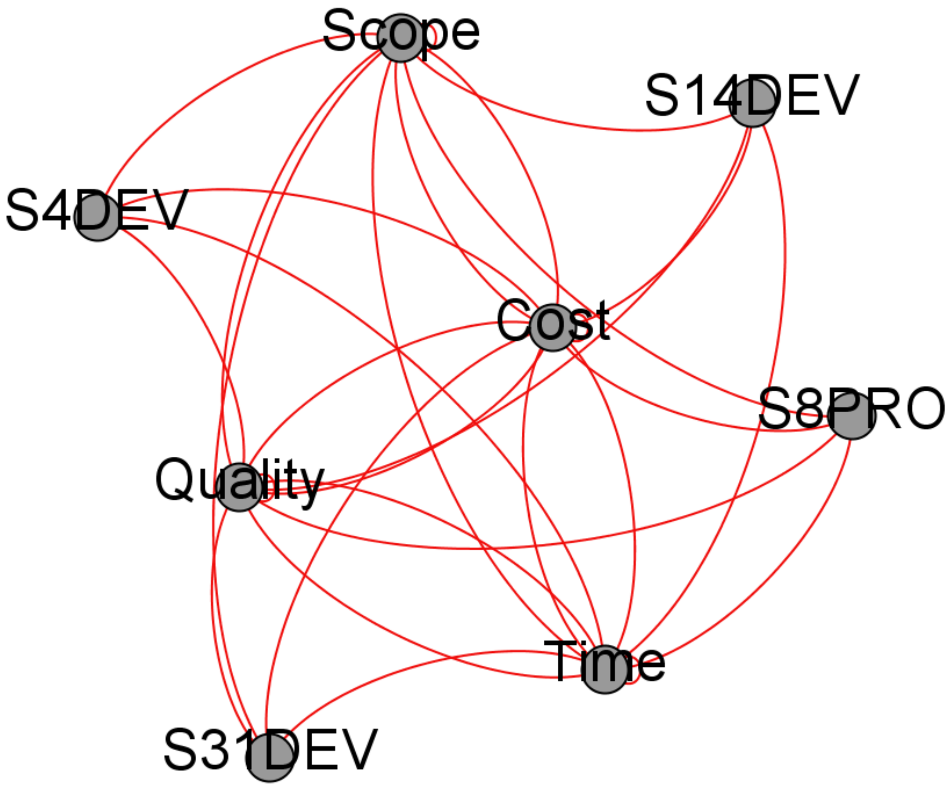


Figure 14-4 The top central factors to all criteria

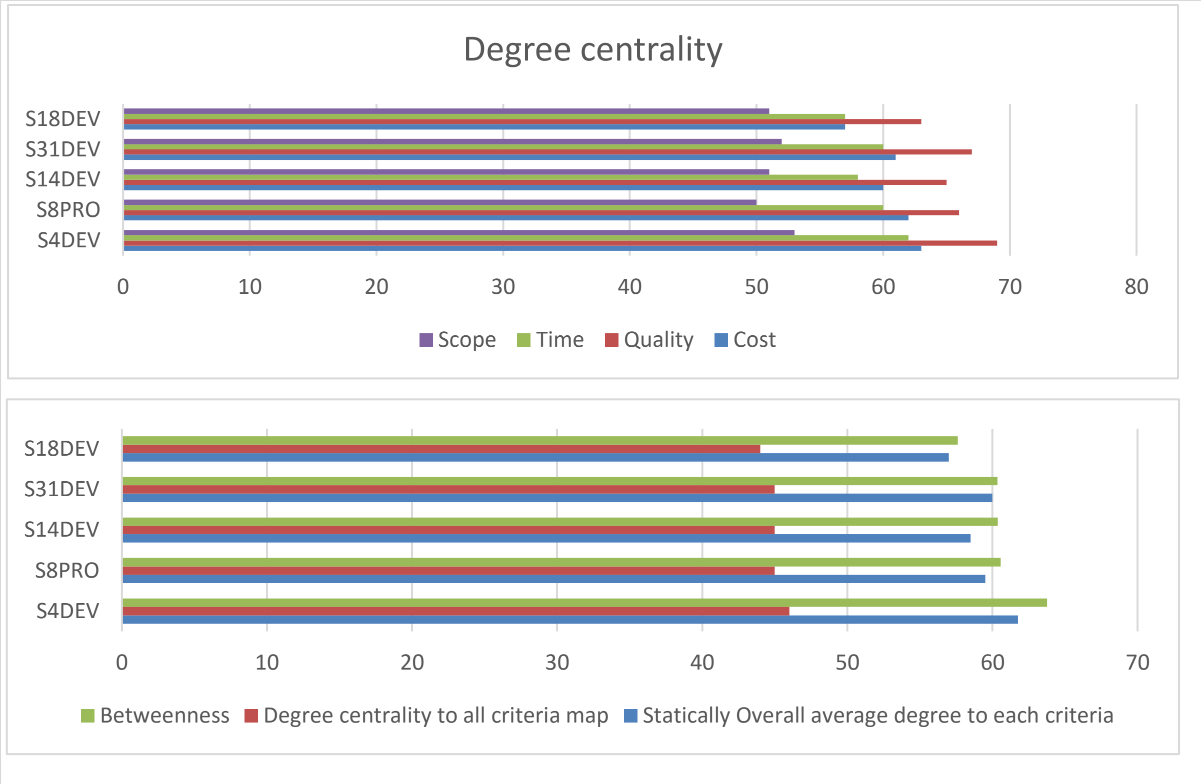


Figure 14-5 Centrality values of the top central success factors to all criteria

This network shows the risk and success factors connected to all four success criteria. Out of 106 possible factors, 80 are connected to all four criteria. From above, it can be seen that the top five central success factors are S4DEV Effective project management, S31DEV Organisational culture, S14DEV Effective monitoring and control, S8PR Realistic budget and S18DEV Change management, whilst the table above shows that S4DEV is the most central factor to all criteria. Software development management is a multi-dimensional task; its importance relates to it being connected to making procedures, and its involvement in planning, organising the project and manging staff. These everyday tasks among others reflect its impact on the success of a project where, according to Vadlamani et al. (2016), “none of the other management activities can yield advantage more than software development from effective project management”. They added: “The efficiency of software project management is reliant on multi-disciplinary, interrelated factors including the management of project range, project time, project budget, project eminence, project human resource project communication, project procurement and project incorporation".

The second factor, S8PR Realistic budget, in the same way affects the software project due to its impact on all parts of the project, as it is involved with people, technology, services and infrastructures, among other things (Newton, 2015). Success in every criterion could be measured in comparison to the amount of money spent on the project, which makes these two factors more important to a software project than the other factors. Although S18DEV Change management is important to the success of a software project, its influence on the project depends on unexpected events. S14DEV Effective monitoring and control and S31DEV Organisational culture are also recognised as being important, but their importance is limited compared to the factors of effective management and budget, as applying good, effective monitoring and control will be limited to the Development class, whilst organisational culture will interact more with humanity and personal concepts about the project (Hofstede et al., 1990). As it according to Yew Wong (2005) "It defines the core beliefs, values, norms and social customs that govern the way individuals act and behave in an organisation".

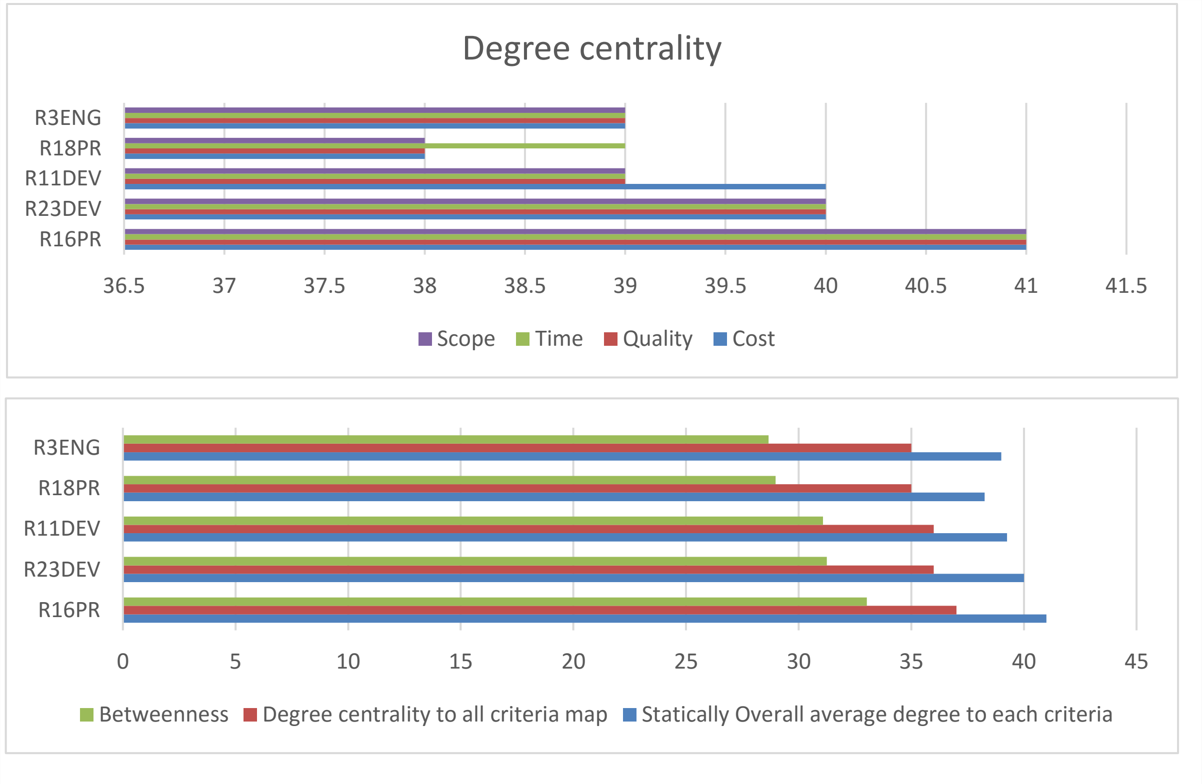


Figure 14-14-6 Centrality values of the top central risk factors to all criteria

From Figure 14-14-6 above, it can be seen that the centrality differences between risk factors to all criteria and success factors are significantly low compared to the noticeable success factors. R16PR Unrealistic budget is the most central factor in this network, threatening the software's success during all phases of the project life cycle (Hijazi et al., 2014a). This shows that there are many factors interacting with and affected by this factor and its ability to define the success or failure of a software development project. Also, this research found that R23DEV Project manager lacks experience is very important to a project's success, which again shows that the management part of the project is responsible for many of the failures in a software project. The project manager is engaged with many sectors like, according to Baccarini et al. (2004), "requirements definition, communication management, human resource management". In addition, staffing, planning and production have also been linked to mangers' experience and can lead to software failure (Basten and Sunyaev, 2014). R3ENG Lack of technical skills, R18PR User resistance and R11DEV Unrealistic resource planning have been recognised as being less influence to a project, which could be because user resistance usually happens at the last stages of the project, whilst technical skills are influence to the quality of a project and in saving time, but are also usually important during the development stage, whilst unrealistic budget and Project manager lacks experience have an impact and influence on other project aspects before, during and at the end of the project. On the other hand, this research argues that there is an almost equally important factor to Unrealistic budget, which is R17PR Resource insufficiency, which has been found to be in the top central factors in three ego networks, cost, quality and time, but, because it is not connected to the scope ego network, it has not been included in all criteria network. The centrality of this factor will be discussed in depth in the next section.

14.4 Research question 3

What influence do risk and success factors have on success criteria?

14.4.1 *Introduction*

In this part, this research discusses the results from the interaction network for each criterion. Section one discusses the results from the regression and modelling of risk factors on the criteria as well as the correlation between the factors while section two discusses the results from the regression and modelling of success factors on the criteria as well as the correlation between the factors.

14.4.1 *Ego network topology used in this research*

Ego network is the second type of the network analysis, where the first type is the complete network analysis (Akhtar, 2014). The ego has been used by many studies to analyse networks from the connectivity and/or characteristics perspectives (Everett and Borgatti, 2005). One of the most famous areas where ego networks are applied is social network analysis. Moreover, according to Arnaboldi et al. (2012), ego social networks are about isolating the individual, who is called the ego, and exploring the inner relationship of this ego in a virtual environment. In addition to that, the ego networks have been used in many subjects like in the biology and diseases field in order to investigated sub-networks of a large-scale biological network in order to prioritise and study the centrality of gene markers in those sub-networks (Yang et al., 2014). Ego networks study the interactions and interrelationships that occur around individual, gene or factor. Therefore, this research uses ego network analyses to isolate the ego network criterion in a virtual environment network created by a matrix of inaction which presents the criteria relationships with the risk and success factors. Also, the ego network presents the influence of those factors on each other on the selected ego criteria. In addition to that, the ego network has the characteristics of showing only the factors that interact with the criteria and eliminates the factors with no influence on the sub-network selected. Another novel aspect of using the ego network is that it shows the centrality measurements of risk and success factors in relation to the selected ego criteria. To conclude, the ego network has proven that it has the ability to study sub-networks from the interaction point of view as, according to Eleta and Golbeck (2012), "The ego network has become a standard unit of measurement for studying small-scale interactions, or micro-sociology", as has been mentioned, which is a very important aspect of exploring complexity as sub-networks and their ambiguous interactions have always been linked with complexity. Thus, a more detailed discussion about the results from the criteria ego network will be discussed in the next sections.

14.4.2 *Cost ego network*

14.4.2.1 *Risk influence on Cost*

14.4.2.1.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to cost	.999[a]	.999	.998	.02185	.999	715.483	59	46	.000

In this network, the cost has been isolated by using the ego topology in order to figure out how the factors are connected and their impact on the cost. Also, the extent of their influence on each other in the network is isolated. Cost is connected to 96 factors out of the 106 factors available; the centrality measures have shown that the top central success factors are S4DEV, S5DEV, S8PR, S31DEV and S3DEV. After applying the degree range topology and including the betweenness centrality, S4DEV Efficient project management is the most central factor in the network. Furthermore, its main impact on cost is due to its ability to connect to the factors related to planning, people and effort as, according to Lee and Yu (2012), efficient project management is defined as "working without waste or using a minimum of time, effort and expense". In addition to its centrality, it has been recognised as the factor with the most control and impact on the risk factors. Furthermore, S4DEV is connected to 58 risk factors. Areas such as communication, leadership, and project-related technology are mainly affected by project management efficiency (Napier et al. 2009, Duncan 1996, Thite 1999, Kirsch 2000).

In the S4DEV network, there was a 99% association between Risk to cost and their risk impact events. that the model showed that an increase in R3ENG Lack of technical skills, R48ENG Inconsistent coding style and R56DEV No backward compatibility and version management plan has the most effect on project cost. This research found it interesting that R44ENG Less code reusability was not one of these factors, although less code reusability will increase the cost of developing a software project as well the cost of maintaining the code (Singaravel, Palanisamy et al. 2010). Even in an organisation with experience in developing software projects extra costs will be added if they do not reuse components (Sandhu and Singh 2006). In addition, Heineman and Councill (2001) have argued that reuse of software components could lower overall costs by 60%. The same can be said about R16PR Unrealistic budget, where the reason behind that could be due to the fact that there are no big differences between the top 20 factors in terms of their impact, as the B-value is between 0.02 and 0.031). For example, R44ENG Less code reusability has the most impact on cost in the S4DEV network as, for every unit increase in R44ENG, a 0.27 unit increase in cost is predicted. This result shows that the huge impact on cost prediction comes from many factors rather than a few selected factors.

14.4.2.1.2 The association between the network nodes

From the correlation, the research has found that there are 18 factors with a correlation that is significant at the 0.01 level and 28 factors where it is significant at the 0.05 level with S4DEV. This result shows that there is a strong a positive relationship between S4DEV and Risk-to-cost . Furthermore, R34ENG Incompatible development environment has the strongest relationship with S4DEV, as identifying the all technical requirements and taking into account all aspects of the development environment has a massive impact on reducing the project risk (Dey et al., 2007), which is another aspect of effective management. An incompatible development environment could lead to a developer having to develop the software ‘from scratch’ (Schottner, 2003), which would affect the project’s overall cost. This explains the strong and positive relationship between S34ENG and Risk-to-cost [r= 0.682, p<.0005]. The results also show that S51PR Data privacy issues has a strong correlation with S4DEV although, in software project according to Ganji et al. (2015), it is hard for people to win in court in relation to data privacy issues. However, in effective management this should be taken into consideration from both a cost and an ethical point of view; not all the people involved in a project should gain access to all of the data: sensitive data should be protected (Saltz, 2015). Also, there is a strong, positive correlation between the risk to cost R52ENG Insufficient consideration of reliability and availability and R48ENG Inconsistent coding styles, as inconsistent coding styles cause programmers to write codes more slowly, which in turn will add extra time and cost to the project (Goodliffe, 2014). Ongoing reliability and availability problems could lead to the extra cost; this can be seen, for example, in the Customer Relationship Management (CRM) software system called the “DIVINE system” where, because of its reliability and availability problems, it had to be replaced (Huijgens et al., 2016).

14.4.2.2 ***Success factors’ influence on Cost***

14.4.2.2.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Success to cost	.999[a]	.999	.998	.02238	.999	1645.238	37	69	.000

This research has found that R16PR, R17PR, R11DEV and R23DEV are the most central risks related to centrality measures. R16PR Unrealistic budget is the most central factor in terms of its connection to the success factors in the cost ego network. Baccarini et al. (2004) ranked unrealistic budget and schedule as the second greatest risks to IT projects. Also, the importance of this factor in terms of its effect on other factors is due to its occurring during the requirement analysis.

In the R16PR network, there is a 99% association between Risk-to-cost and their risk impact events. S8PR Realistic budget has the most effect on project Risk-to-cost as, for every unit of increase of S8PR, there is a 0.042 unit increased in the success-to-cost . The association between success-to-cost and S8PR Realistic budget is due to the fact that a realistic budget is an essential factor in cost control management (Ruhe and Wohlin, 2014a). Its effect on cost also comes from the large number of factors it affects, for example, “travel, meetings, computing resources, software tools, special testing and simulation facilities, and administrative support”(Sajad et al., 2016). This research believes that the reason behind the low association of S3DEV Realistic schedule with success-to-cost could be because project schedules in Saudi Arabia are not considered to have a huge effect on the project cost due to their changing nature, and the results have recognised that S33DEV Project criticality has more effect on cost.

14.4.2.2.2 The association between the network nodes

The results show that 11 factors are significantly correlated with R16PR. Five factors have a correlation that is significant at the 0.01 level and six have a correlation that is significant at the 0.05 level. It is understandable to find that S8PR Realistic budget has the strongest association with R16PR Unrealistic budget, although this research has noted that this relationship between the two factors should be a negative relationship as having a realistic budget should eliminate or reduce the risk of having an unrealistic budget. However, the main reason behind this positivity of relationship is due to the design of the question itself as it is looking at the impact of the risk and software factors on the success criteria rather than looking at their impact on each other. In addition, realistic budget has been found to have a strong and positive relationship with Success-to-cost which indicates that is has more association with success-to-cost that R16PR. The association between R16PR and Success-to-cost is not one of the strongest associations; this research argues that its indirect correlation with success-to-cost is what makes it important to success-to-cost as it has an association with most of the success factors having a significant impact on success-to-cost . The results also show that there is a strong, positive correlation between Success-to-cost and S11DEV Appropriate development process/methodology. This result agrees with what Verma et al. (2014) have noted: that in software development different methodologies will have different cost effects on the project cost.

14.4.3 *Quality Ego network*

14.4.3.1 *Risk to quality*

14.4.3.1.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to QUALITY	1.000a	1.000	1.000	.00306	1.000	30916.590	65	41	.000

In the S4DEV network in the risk to quality network, the sub-model is associated with a large number of risk impacts. There is a 100% association between Risk to QUALITY and their risk impact events, which reflects the influence of S4DEV on the risk factors’ interaction with success. Furthermore, all the risk factors connecting risk to quality are connected to S4DEV. The research results have shown that R17PR has the largest beta coefficient of 0.178, which means that this factor has the strongest unique contribution to explaining the project QUALITY. The risk-to-quality prediction increased by 0.178 for every unit increase in R17PR Resource insufficiency. For example, insufficient resources may result in software developers reducing the quality of the code and the time needed for development to match the available but insufficient resources. Also, it may affect the efficiency of the software, and efficiency is considered one of the main quality factors (Addison and Vallabh, 2002). Resource policies play a significant rule in software project quality (Thakurta and Square, 2012), although it is worth mentioning that there is a lack of studies on resources policies. According to Otero et al. (2009), this could be due to the fact that most development and management teams believe that every software project has its own unique scenario.

14.4.3.1.2 The association between the network nodes

The results show that 24 factors are significantly correlated with S4DEV. Half of those factors have a correlation that is significant at the 0.01 level and 12 factors have a correlation that is significant at the 0.05 level. R14DEV Inefficient team capabilities has the strongest and most positive relationship with S4DEV of all the risk factors [r= 0. 448, p<.0005]. This result shows the importance of having good management of team capabilities as, over the years, good team capabilities have been proven to be more important to the success of software projects than good tools (Trendowicz and Jeffery, 2014b). It is interesting to find that factor R17PR Resource insufficiency has the second strongest relationship with S4DEV, which shows two important characteristics of a node., Its position in the network, as it has been recognised as the most central node for the risk-to-quality network, and its important impact on the most central node in the Success-to-quality network, S4DEV. R34ENG Incompatible development environment has the strongest and most positive relationship with risk-to-quality [r= 0. 697, p<.0005] as an incompatible development environment impacts on quality due to its limited ability to develop certain applications for the software project. Its impact on the software project quality has also been recognised by Dey et al. (2007), although in their case study they did not find this factor to have the most impact on the overall project success. R64DEV Project distribution also has a positive and strong relationship with project QUALITY [r= 0. 689, p<.0005], although Bird et al. (2009) have concluded that there is no significant association between geographical distribution of projects and software quality in their studies about the Windows Vista projects. However, Cataldo and Herbsleb (2011) found in their study that failure rates in geographically dispersed projects are 14 times higher than in projects where developers are working in the same place. Furthermore, Cataldo and Nambiar (2009) in another study found that there is a significant association between the distances between the developers and the number of defects, as the greater the distance the greater the number of defects. Furthermore, Ramasubbu et al. (2011) found a significant association between the number of defects and the geographical distribution of projects in their study of 362 projects (Nguyen-Duc et al., 2015). This leads to the conclusion that there is a significantly strong association between geographical distribution of projects and quality. Saudi Arabia is not an exception in this matter as, according to Jager et al. (2008), "The coding and the other parts of the design and testing are often done offshore".

14.4.3.2 ***Success to quality***

14.4.3.2.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Success to QUALITY	1.000[a]	1.000	.999	.01389	1.000	3684.780	38	68	.000

In relation to the success to quality, the results show that R17PR Resource insufficiency is the most influential factor, as it is connected to 37 factors and has the most centrality measures. In the R17PR network, S24DEV Clear assignment of roles and responsibilities has the most contribution to project QUALITY with a value of 0.0374. This shows the impact of clarity in attaching roles and responsibilities to the development team as a lack of clarity could have an impact on the project quality if there were any misunderstandings about responsibilities during a project, or in some cases it could lead to conflict between team members. Schüler et al. (2015) went to the extent of stating that it is one of the most important factors in a successful quality control process, because it is an ongoing process. Furthermore, the clear assignment of roles and responsibilities helps to increase effectiveness and quality, as Barney et al. (2014) noted that stakeholders in a software project must have a clear understanding of their roles in the project, as this type of clarity could reduce the risk of project failure. It is interesting to note that the S29DEV Team capabilities has relatively the second strongest contribution to quality, as it has also been recognised as having a positive and strong contribution to the project cost, which shows this factor's overall contribution to project success.

14.4.3.2.2 The association between the network nodes

There are 18 factors that are significantly correlated with R17PR. Thirteen factors have a correlation that is significant at the 0.01 level and five factors have a correlation that is significant at the 0.05 level. S8PR Realistic budget has the strongest association with R17PR Resource insufficiency [r= 0. 422, p<.0005]. It is important to have a realistic budget as the majority of risks facing a software project during the project, according to Sipayung and Sembiring (2015), come from resource insufficiency and functional problems. S4DEV has the second strongest association with R17PR Resource insufficiency as, in dealing with resource inefficiency, a good project management team has the ability to improve the overall project quality. In addition to its centrality to the quality network, resource insufficiency has success-to-quality as one of the top five associations, which shows the importance of its physical location and impact on success-to-quality . With regard to success-to-quality , it has the strongest relationship with S15PR Adequate resources. It is not surprising to find that this factor has the strongest relation with success-to-quality as resources has been the controlling factor in the quality network, so it is understandable to find that resource insufficiency is the most central factor to the success factors where the adequate resources factor has the strongest positive association with success-to-quality . Moreover, S24DEV Clear assignment of roles and responsibilities has the second strongest relationship with Success-to-quality , which adds to its importance as the factor making the greatest contribution in success-to-quality , as the results have shown in the regression of success-to-quality .

14.4.4 *Time Ego network*

14.4.4.1 ***Risk influence on time***

14.4.4.1.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to TIME	.998[a]	.996	.992	.03645	.996	218.732	58	47	.000

Once again, S4DEV has also proven its importance to risk-to-time , as in its network with risk-to-time there is a 99% association between Risk to TIME and their risk impact events. One of the variables included in the model that makes the greatest contribution to the prediction of risk-to-time found by this research is R9DEV Inadequate infrastructure. For every unit increase in R9DEV a 0 .0354 unit increase in TIME is predicted. The main risk is Inadequate infrastructure. This research believes that the issue it poses to software project time is the number of elements that it contains and the ability of those elements to delay the overall project. According to Nidhra et al. (2013), those infrastructure elements are “dependable electricity supply and alternate power sources, adequate telecommunication infrastructure including dependable internet connection and bandwidth”.

Two points have been noticed about this network and how the factor contributes to risk-to-time . First, all the risk factors in this multiple regression have a positive impact on TIME, except for R10DEV and S4DEV. Second, it is also noticeable that R10DEV Unrealistic schedule has a beta value of -.001, which does not reflect its impact on the project Risk-to-time as, according to Alenezi et al. (2015), an unrealistic schedule is considered one of the top ranked factors in a project's failure. Also, Sonchan and Ramingwong (2014) noted that a major factor in not delivering a software project on time is the project's unrealistic schedule (Hoodat and Rashidi, 2009; Yu, 2011; Han and Huang, 2007). This could be due to two elements. First, the fact that all factors have been considered to have a significantly low contribution to risk-to-time where the strongest as has been mentioned has a contribution value of less than 0.04. Second, it could be because project delivery time is not considered a big issue in Saudi Arabia, especially in government or semi-government sectors.

14.4.4.1.2 The association between the network nodes

There are 14 factors that are significantly correlated with S4DEV. Eight have a correlation that is significant at the 0.01 level and six have a correlation that is significant at the 0.05 level. R23DEV Project manager lacks experience has the strongest and most positive relationship with S4DEV of all the risk factors. Many project managers in the technological are promoted when they show good technical skills rather than good management skills (Palm and Lindahl, 2015). Savelsbergh et al. (2016) have also argued that project managers tend to learn from their experience rather than what they have studied. This type of skill is important as it could lead to efficient project management. It is noticeable that the correlation should be a negative rather that a positive association as the more experience a manger has, the more effective is her/his management of the project. In relation to risk-to-time , the strongest correlation is with R57ENG Use of cheap tools; this shows the importance of not using cheap tools in relation to project time. Gea et al. (2011) noted with regard to software engineering that: "Cheap tools don't deliver sophisticated features". Programmers tend to spend time modifying and manipulating codes when they use cheap or free tools like free compilers instead of buying the full features available (Koopman, 2010). Although there is no doubt that it could save money for the project in the short term, in the long term it could affect the project delivery time. Also, R60DEV No training for managing outsource relationships has a noticeable strong and positive association with risk-to-time . Managers without training to deal with factors like “Language barriers in project communications, Constraints due to time-zone difference” (Nakatsu and Iacovou, 2009) could have a catastrophic effect on the project time delivery and the overall project success.

14.4.4.2 *Success factors’ influence on time*

14.4.4.2.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Success to TIME	1.000[a]	.999	.999	.01545	.999	2620.123	38	68	.000

In R17PR there is a 99% association between SUCCESS-TO-TIME and its success impact events. S33DEV Project criticality makes the most contribution to project success-to-time . Furthermore, an increase of unit in project criticality increases the success-to-time prediction. Criticality of the project is very important in case of unexpected events happening as Cohen et al. (2004) argue that project criticality is a major factor in determining the process chosen in the software development which have a major impact on the project time. In addition, project control becomes more effective in a project with high criticality, especially if the project is using the traditional development methodologies (Cockburn, 2006). Furthermore, project criticality should be a very important factor when the project methodology is chosen (Attarzadeh and Ow, 2008). S13DEV Up-to-date progress report and S36PR Commitment of stakeholders have relatively the second strongest contribution to success-to-time . This result shows the importance of the project report documentation to the project success-to-time where an event like in case of programmers Leaves or resigns from the project but there an up to date progress report will save time and help the replace programmer and the development team as the report should have a slandered used in the development software project or in some cases the coding style used in the development. In turn, having committed stakeholders has a positive impact on the project time because this is an important factor in encouraging the users' involvement, especially in the requirement stage (Zowghi and Coulin, 2005) and especially if they are "unfamiliarity with the software development field" (CONSTANTIN and MANURASADA, 2015).

14.4.4.2.2 The association between the network nodes

There are 21 factors that are significantly correlated with R17PR. Fifteen factors have a correlation that is significant at the 0.01 level and six factors have a correlation that is significant at the 0.05 level. There was a strong, positive correlation between the R17PR Resource insufficiency and S8PR Realistic budget. This significant association agrees with the theory of mental budgeting, in which mental theory has links between limited resources and budget (Lee et al., 2012). In a software project, a realistic budget will be negatively affected by resource insufficiency, although the mental theory has explained that people tend to not ask for extra funds in this scenario but they will not be able to deliver the product to the time, cost and quality wanted in most cases. Lee et al. (2012) have also noted that resources are usually highly associated with budget and schedule. Change management should be useful in the case of resource insufficiency, which explains why S18DEV Change management has the second strongest and most positive relationship with success to R17PR. The results also show that there is a strong, positive correlation between SUCCESS-TO-TIME and S16DEV Good leadership, where good leadership could be one of the most important skills in a project's success (Hadad et al., 2013). It is very important to lead the team in completing the project on time as well avoiding any obstacles facing it, especially if these are related to the project team where, according to Keil et al. (2013), it is ranked as the most important skill for a manager to possess. Also, a major aspect of good leadership's effect on project success is in: "providing a vision, demonstrating charisma, and leading by motivating people toward accomplishing the project goals"(Keil et al., 2013). It is understandable why S10DEV Familiar with technology has the second strongest relationship with SUCCESS-TO-TIME as familiarity with technology usually saves time. For example, if the programmers are not familiar with the technology, this adds time because prototypes have to be designed to ensure that the technology has the capability to deliver what it is going to be used for, especially in the most risky parts of the software (Zaied et al., 2013).

14.4.5 *Scope ego network*

14.4.5.1 *Risk influence scope*

14.4.5.1.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Risk to SCOPE	.998[a]	.996	.993	.04771	.996	309.312	49	57	.000

There is a 99% association between risk-to-scope and its risk impact events in the S4DEV network. This research found that R54DEV No update plan to the final software product has the largest beta coefficient of 0 .04, where for an increase of a unit in . R54DEV an increase in RISK-TO-SCOPE is predicted. Its risk impact comes from the relative importance of preparation like "threat models, design specs, test plans, and code analysis tools" (Dadzie, 2005) where, if the software has no update plan in case of software upgrade, it could lead to new features being added out of scope or it could pose new security threats to software, as 'patching', also known as 'aggressive software update plan', is the most suitable solution against that threat (Basham and Rosado, 2005). Also, it is understandable that R22DEV Improper planning has the second greatest contribution to risk-to-scope as, according to

Ahimbisibwe et al. (2015), having a pre-specified plan is one of the scopes determined during the development stages. Also, it is noticeable that there is no one factor that makes a significant contribution to risk-to-scope . Instead, this research believes that a group of factors have a significant impact on risk-to-scope . In addition, this research has found that, in the top 10 factors that contribute to risk-to-scope prediction, six are from the development class, where an increase in a unit in the development class has an increase of 0.499 units in the risk-to-scope prediction. It is interesting that R1ENG Unclear customer requirements and R2ENG Unable to meet user requirements are not found here as one of the contributory factors to rscope, as this research believes that those factors should have some sort of contribution to the software project scope as they help to define the content of the project.Tomer (2014) agrees with this research on the importance of those factors, as he has noted that “project scope management is concerned with collecting the requirements from stakeholders”.

14.4.5.1.2 The association between the network nodes

There are 38 factors that are significantly correlated with S4DEV Effective project management. Thirty-one have a correlation that is significant at the 0.01 level and seven have a correlation that is significant at the 0.05 level. R34ENG Incompatible development environment has the strongest and most positive relationship with S4DEV. Developing software with the same features and specifications required by the customer is one of the definitions of scope that effective managers must ensure (Kumari and Pillai, 2014). It is becoming increasingly necessary that management choose the programming language that can deliver the features specified. It is interesting to see that there is a strong, positive correlation between the risk SUCCESS-TO-SCOPE and R57ENG Use of cheap tools. Kumari and Pillai (2014) noted that project scope is measured by providing the product with features and functions as planned. As a result, cheap tools usually have a limited ability to provide a product with the agreed features (Gea et al., 2011). Thus, it is understandable to find it has a major association with risk-to-scope . This again shows the importance of this factor to the overall success of a software project as it has also been recognised as having a strong association with the time criterion as well. R31ENG Improper marketing techniques also has a positive and strong relationship with project RISK-TO-SCOPE . Marketing is one of the key factors that is important to users as well as helping to manage their expectation of the software product (TAHERDOOST and KESHAVARZSALEH, 2015a). In some software projects, if the project presents the wrong features, functions or even the wrong potential customers, it could have a catastrophic effect on the project's planned scope. It is noticeable that many unsuccessful software projects have failed due to improper marketing techniques (Asif et al., 2014). It is also strange to find that R1ENG Unclear customer requirements has not been recognised as having a significant correlation with risk-to-scope . The main reason behind that could be because many software projects are developed in advance, and then rely on marketing, and so could have a good return on investment.

14.4.5.2 *Success factors' influence on scope*

14.4.5.2.1 Modelling of the relationship between the isolated network nodes

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Change Statistics				
					R Square Change	F Change	df1	df2	Sig. F Change
Success to SCOPE	1.000[a]	.999	.999	.01826	.999	3602.919	37	69	.000

In the R16PR network, there are is a 99% association between SUCCESS-TO-TIME and its success impact events. S37DEV Team training makes the biggest contribution to project SUCCESS-TO-SCOPE with a value of 0.054, as an increase in a unit of S37DEV increased the success-to-scope prediction by a unit. It is interesting to find a certain three factors are not part of the top contribution to predict success-to-scope , as these factors – S1ENG Clear requirements and specifications S2ENG Clear objectives and goals and S18DEV Change management – play a critical role in managing the scope creep. Keil et al. (1998), Thakurta (2013) agreed that clear software goals and outcome reduce the probability of having an out of scope expectation and objectives. Also, in most software projects there is some sort of scope shift; it is in the hands of the project managers to apply good change management so they can steer the project to be completed for the cost available and within the agreed time, as well as ensuring that the customers are satisfied with the final product. This result could be due to team training, which could improve the team's ability to deal with scope creep. Santos et al. (2013) mentioned four areas of improvement in software development in agile methodology, and team abilities was one of those areas.

14.4.5.2.2 The association between the network nodes

There are 33 factors that are significantly correlated with R16PR. Twenty-four factors have a correlation that is significant at the 0.01 level and nine have a correlation that is significant at the 0.05 level. There is a strong, positive correlation between the R16PR Unrealistic budget and S8PR Realistic budget, which shows their importance and the relationship between them, although this research has noted that this relationship must be a negative one. This is because a high association value of realistic budget has a low association value of unrealistic budget. The reason behind this association positivity has been explained. S37DEV Team training has the second strongest relationship with success to R16PR, as team training has been found to be very useful in reducing the project duration time, which is also a very important factor on the overall cost of the project. The results also show that there is a strong, positive correlation between SUCCESS-TO-SCOPE and S16DEV Good leadership, as good leadership is one of the important 'soft' skills that a software management team should have in order to achieve the planned project goals as well as keep the developers motivated and focused on the project scope (McLeod and MacDonell, 2011). Keil et al. (2013) have also found that leadership is the top ranked skill in software project success. One of the good practices that leaders should carry out is to make plans to manage the project scope. Furthermore, according to Keil et al. (2013), leadership is all about “providing a vision, demonstrating charisma, and leading by motivating people toward accomplishing the project goals”, all of which has the ability to keep the project within the project scope as well establishing good communication between managers and stakeholders. S35ENG Extensive testing for quality has been found to have the second strongest relationship with scope. This result again shows that link between project scope and overall project quality, as testing a product for quality usually reveals if the future of the product is within the planned project scope.

14.5 **Summary**

This chapter was divided into three sections to discuss the research results. The first section discussed the most important risk factors in a software project, dividing the factors into three classes: Product Engineering, Development Environment and Program Constraints. The second section was divided into two parts. In the first part on the construct correlation, the most central factors in the sub-networks have been discussed. In the second part, the most central factors for all success criteria from the construct correlation were discussed. The third section reviewed and discussed the influence of the risk and success factors on success criteria through discussing the influence of a risk factor on success factors in the cost ego network, by discussing the contribution of all factors connected with this risk factor in prediction of cost. In addition, this research discussed the relationship between these factors. Additionally, the success factors’ influence on cost has also been discussed in the same way. Furthermore, the same methodology has been applied for quality, time and scope for the purpose of answering the research question. The next chapter will present the research conclusions.

Chapter 15: Conclusion

Chapter 15: Conclusion

15.1 Introduction

This chapter presents the research conclusions. First will review the objectives achieved by this thesis by linking the objectives with the relevant chapters. After that, the research limitations will be identified. Third presents the contribution of this research to the knowledge. The chapter ends with the research recommendations and proposals for further work.

15.2 Accomplishing the research objectives

15.2.1 *To review, extract and classify risk and success factors in software development projects*

Software projects have been proven to be one of the main types of project that fail completely or partially. To achieve the objective relating to the review, extraction and classification of risk and success factors in software development projects, in Chapter 3 the research defined risk in software projects. It discussed that 60% of software projects fail to achieve their set targets (GROUP, 2013, Raith et al., 2013). In addition, this research performed a systematic review of the available publications on the risk in order to extract factors that could have a negative impact on software development, and a review of the existing classifications of risk in IT and software development projects. Through this systematic review, the SEI taxonomy was chosen as the classification to be used in this research. The SEI taxonomy is used by many companies, institutions and organisations in both the public sector and the private sector to identify and classify risk in software development projects. In the same way, this research has also reviewed and defined the general concept of success in software projects as, in order to achieve this objective, this research had to identify the criteria that measure the success of a project, which are cost, quality, time and scope. In addition, from the literature review, this research has extracted the important factors that have a positive impact on software projects; these elements were classified according to the taxonomy used in this study.

15.2.2 *To analyse the influence of risk on success factors and criteria using network metrics*

To achieve this objective, in Chapter 4, using the literature review, this research identified betweenness, closeness, degree and eigenvector centrality as tools to study the interactions and influence of network nodes. In addition, the meaning of each metric was interpreted and illustrated in relation to the research aim and objectives. The results from the construct correlation were used to build a network of interaction, as has been explained in Chapter 5 in order to study the influence occur between the research constructs. Furthermore, through the construct correlation this research analysed the influence that occurs between risk and success factors and success criteria by studying each network separately. It statistically analysed the influence of risk factors on success criteria, influence of risk on success factors' network characteristics, and the influence of success factors on success criteria network characteristics in section one of Chapter 8. Section two of Chapter 8 analysed and ranked the most central factors in each of three networks by identifying the top five central factors for each of the four centrality measures.

15.2.3 *To measure (capture) the interdependency between risk and success factors and criteria*

To achieve this objective, and through the extensive literature review, this research created a dependency matrix through the creation of a questionnaire for the experts to determine the interactions that occur between the elements of software development projects. After that, the network analysis program Gephi was used to create a network of interaction, as has been described in Chapter 4. The network showed the interactions that occur between the risk factors, success factors and success criteria. Chapter 9 measured and analysed the relationships between the factors connected with all the success criteria – cost, quality, time and scope. Moreover, 80 factors were found to be directly connected to these criteria. This research also studied interdependence between research constructs via a review of the results of the five central factors in the network. This analysis was divided into two parts. In the first part, the top success central factors for all criteria were analysed. In the second part, the top central risk factors were also identified and analysed. In both parts, the centrality results were reviewed in terms of impact on each criterion, ranking, degree and betweenness. In addition, overall average degree results and ranking of the factors in the network were compared statistically, as described in chapters 9 and 14.

15.2.4 *To use the ego topology to measure the influence of risk and success factors on success criteria*

After the network as a whole was explored and analysed by this research in chapters 8 and 9, and in order to achieve this objective of studying the network micro-sociology, the ego topology was used, because it focuses on studying a selected node environment and the nodes that interact with the ego node (Ortega, 2014). This research used ego topology with cost in order to identify factors that are connected directly with cost without the intervention of the other criteria. In Chapter 10, the top central success and risk factors in this criteria were identified. Furthermore, the same methodology was applied to the quality ego network, as can be seen in Chapter 11, time ego network in Chapter 12 and scope ego network in Chapter 13.

15.2.5 *To isolate latent success factors and their associations with risk factors*

After studying the whole network and ego criteria network, this research identified and studied the success factor that had the most influence on the risk factors. To achieve this objective, the research used the ego topology in the cost network, as illustrated in Chapter 5. S4DEV Effective project management was identified as the most central and influential factor in this network, and all risk factors directly connected by this factor were selected. Furthermore, the construct correlation results were used to measure the impact of those factors, and statistical regression was used in prediction of the cost. After that, this research studied the relationship between these factors through the use of statistical correlation, as was described in Chapter 10. The same methodology was applied to identify success factor that had the most influence on the risk factors in the other three ego network criteria. Similarly, effective project management was found to be the most influential factor in the quality ego network, as seen in Chapter 11, time ego network in Chapter 12 and scope ego network in Chapter 13. All the results were discussed in Chapter 14.

15.2.6 *To isolate latent risks that influence success factors*

To achieve this objective, this research showed the progression that occurs when changing degree range in ego criteria. Chapter 10 found that the most influential risk factor on success factors is R16PR Unrealistic budget in cost ego network. In order to study its influence, all the associated factors with cost and R16PR were isolated. Moreover, the relationships between all the factors were examined, as well as the extent of the participation of each factor in the cost prediction through the use of correlation and regression. The same methodology was used in the quality ego network and time ego network in chapters 11 and 12 respectively. Furthermore, R17PR Resources insufficiency was identified and isolated to study its influence in both ego networks. In scope ego network, R16PR Unrealistic budget was identified as the factor with the most influence on success factors, as was analysed in Chapter 13. All the results were discussed in Chapter 14.

15.3 Research limitations

Any scientific research might face challenges and limitations. For example, it is always going to be linked to a specific time frame. This research, despite its significant contribution to knowledge, is no exception from those limitations, as it faced a number of challenges and limitations as follows:

- Number of respondents: despite the fact that this research compared the number of respondents with published studies and research, and the data collected is considered to be significant enough to create a piece of scientific research, the final number of participants is relatively small. A bigger number of participants would confirm the results reached by this research.
- The search focused on the study of software projects as a whole and did not focus on one type of project or methodology, whereas a project contains many development methodologies , such as SDLC and Agile.
- The data collected did not relate to an existing project; the opinions of the target sample were gathered through their experience with previous projects.
- One of the biggest challenges faced by this research is the questionnaire size. Due to the nature of this research and its objective to answer the research questions, the number of risk and success factors was noticeably large, 102 factors. Many participants started the questionnaire but did not complete it and so they were excluded.
- Repetition: the research design forced the researcher to repeat the methodology and the writing style in relation to the ego networks for cost, quality, time and scope. However, this repetition was necessary to reach the research goal and answer the research questions.

15.4 Knowledge contributions

This research sought to understand the complexity of interaction that occurs in software projects. It contributes to the methodology through linking networking analysis with statistical analysis via use of both primary and secondary data. Its results will provide decision-makers with information that will assist them in making the right decision in order to increase the success of a software project. The following are the key points that have been contributed to knowledge and that helped to answer the research questions:

- Most previous research examined the software project through the perspective of either risk or success via their impact on the success or failure of the project. In contrast, this research has studied the software project through a combination of both aspects and endeavoured to understand the relationship that occurs between these components. As far as the author is aware, this is among the first study to investigate the complexity of interaction between the software project components.
- Using a literature review, this research has extracted and classified a list of risk and success factors that affect the software project.
- This research has presented a visualisation network about the interaction between the software development components.
- It has increased understanding of the interdependence between the software factors of the project by integrating network analysis, regression and correlation.
- This research has explored each success criterion, cost, quality, time and scope, individually in order to understand the interrelationship and influence of the software project components that occur in each one.
- This research extracted the major risk factors that need to be considered by software project decision-makers and stakeholders.

15.5 Recommendations for further research

This research has studied the complexities of the software project in many ways. It focused on uncovering the complexity of interactions and interdependence between the parts of the project, which this resulted in a list of contributions to knowledge, as has been stated above. However, there are several areas that this research believes need to be investigated in future research, as follows:

- This research has studied the technical project as a whole. Therefore, it is recommended that future studies focus on the study of complexity in relation to specific methodology.
- Despite the in-depth investigation conducted by this research about the influence that occurs between risk and success factors, this research recommends studying the internal influence that occurs between risk factors and, in the same way, the internal influence between the success factors.
- By using the results of this, this research, recommends trying to set up a risk assessment tool which can make an assessment of the project software and its ability to succeed.
- Also, this research recommends that this project be based on the results that have been reached to establish a framework for decisions and strategies that can be made when establishing a software project, which can be used in the requirement analysis stage.
- This research recommends that data be collected from a real-life project in order to validate the results that have been reached in this study due to the lack of empirical studies in this area.
- In order to generalise the findings of this research, it is recommended that respondents from multiple countries like the UK and USA should be surveyed, as the results of that should show if respondents from different geographical areas will have significantly different views from those in this study. Moreover, this research recommends that the

demographic background of the respondents like their professional experience age and organisational size should be included and examined to identify if they have an impact on the respondents’ views.

- This research also recommends that the findings of the factors that have the most influence on the cost, quality, time and scope as well as the centrality of those factors should be used in the customisation of the software development methodologies. Furthermore, this research recommends, taking into consideration its results, that projects that use an agile methodology should set up a risk action plan for each development cycle meeting in order to be able to avoid any unplanned decision that could cause the software project to fail. In addition to that, at the end of the development cycle an assessment of the risk factors that accrued and the effectiveness of the decision used to eliminate or minimise their impact should be re-evaluated in order to be able to update the action plan for the next development cycle.

References

AAPOR 2004. Standard definitions: Final dispositions of case codes and

outcome rates for surveys. *Lenexa, KS: AAPOR*.

ABDELRAFE, E. & BURAIRAH, H. 2013. Managing software project risks (Design Phase) with proposed fuzzy regression analysis techniques with fuzzy concepts. *International Review On Computers And Software (I. RE. COS),* 8**,** 2601-2613.

ACKERMANN, F., HOWICK, S., QUIGLEY, J., WALLS, L. & HOUGHTON, T. 2014. Systemic risk elicitation: Using causal maps to engage stakeholders and build a comprehensive view of risks. *European Journal of Operational Research,* 238**,** 290-299.

ADDISON, T. & VALLABH, S. Controlling software project risks: an empirical study of methods used by experienced project managers. Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology, 2002. South African Institute for Computer Scientists and Information Technologists, 128-140.

AGARWAL, N. & RATHOD, U. 2006. Defining 'success' for software projects: An exploratory revelation. *International journal of project management,* 24**,** 358-370.

AHIMBISIBWE, A., CAVANA, R. Y. & DAELLENBACH, U. 2015. A contingency fit model of critical success factors for software development projects: A comparison of agile and traditional plan-based methodologies. *Journal of Enterprise Information Management,* 28**,** 7-33.

AKBAR, S., KHAN, S. U., ALAM, A. U. & ALAM, S. U. 2012. Portfolio Cost Management in Offshore Software Development Outsourcing Relationships from Vendor's Perspective: a Systematic Literature Review Protocol. *IOSR Journal of Computer Engineering (IOSR-JCE),* 6**,** 14-20.

AKHTAR, N. Social network analysis tools. Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on, 2014. IEEE, 388-392.

AL-KARAGHOULI, W., ALSHAWI, S. & ELSTOB, M. 1999. Enhancing the understanding of customer requirements in business software engineering. *University of Westminster, Brunel University*.

ALOINI, D., DULMIN, R. & MININNO, V. 2007. Risk management in ERP project introduction: Review of the literature. *Information & Management,* 44**,** 547-567.

ALTAHTOOH, U. A. & EMSLEY, M. W. Is a Challenged Project One of the Final Outcomes for an IT Project? 2014 47th Hawaii International Conference on System Sciences, 2014a. IEEE, 4296-4304.

ALTAHTOOH, U. A. & EMSLEY, M. W. Is a Challenged Project One of the Final Outcomes for an IT Project? System Sciences (HICSS), 2014 47th Hawaii International Conference on, 2014b. IEEE, 4296-4304.

ALTUWAIJRI, M. M. & KHORSHEED, M. S. 2012. InnoDiff: A project-based model for successful IT innovation diffusion. *International Journal of Project Management,* 30**,** 37-47.

AMARATUNGA, D., BALDRY, D., SARSHAR, M. & NEWTON, R. 2002. Quantitative and qualitative research in the built environment: application of "mixed" research approach. *Work study,* 51**,** 17-31.

ANAND, S. & CHOPRA, V. 2012. Decision Support System for Software Risk Analysis During Software Development. *International Journal for Science and Emerging Technologies with Latest Trends,* 2**,** 29-35.

ANTHOPOULOS, L., REDDICK, C. G., GIANNAKIDOU, I. & MAVRIDIS, N. 2016. Why e-government projects fail? An analysis of the Healthcare. gov website. *Government Information Quarterly,* 33**,** 161-173.

ARLENE, F. & KOSECOFF, J. 1985. How to conduct Surveys. Sage Publications Inc.

ARNABOLDI, V., CONTI, M., PASSARELLA, A. & PEZZONI, F. Analysis of ego network structure in online social networks. Privacy, security, risk and trust (PASSAT), 2012 international conference on and 2012 international confernece on social computing (SocialCom), 2012. IEEE, 31-40.

ASIF, M. & AHMED, J. Analysis of Effectiveness of Apriori and Frequent Pattern Tree Algorithm in Software Engineering Data Mining. 2015 6th International Conference on Intelligent Systems, Modelling and Simulation, 2015. IEEE, 28-33.

ASIF, M., AHMED, J. & HANNAN, A. Software Risk Factors: A Survey and Software Risk Mitigation Intelligent Decision Network Using Rule Based Technique. Proceedings of the International MultiConference of Engineers and Computer Scientists, 2014.

ATKINSON, R. 1999. Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International journal of project management,* 17**,** 337-342.

ATKINSON, R., CRAWFORD, L. & WARD, S. 2006. Fundamental uncertainties in projects and the scope of project management. *International journal of project management,* 24**,** 687-698.

ATTARZADEH, I. & OW, S. H. 2008. Modern Project Management: Essential Skills and Techniques. *Communications of the IBIMA,* 2**,** 1-9.

BABAR, M. A., VERNER, J. M. & NGUYEN, P. T. 2007. Establishing and maintaining trust in software outsourcing relationships: An empirical investigation. *Journal of Systems and software,* 80**,** 1438-1449.

BACCARINI, D. 1996. The concept of project complexity—a review. *International Journal of Project Management,* 14**,** 201-204.

BACCARINI, D., SALM, G. & LOVE, P. E. 2004. Management of risks in information technology projects. *Industrial Management & Data Systems,* 104**,** 286-295.

BAGGIO, R., SCOTT, N. & COOPER, C. 2010. Network science: A review focused on tourism. *Annals of Tourism Research,* 37**,** 802-827.

BALAJI, S. & MURUGAIYAN, M. S. 2012. Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management,* 2**,** 26-30.

BANNERMAN, P. L. 2008. Risk and risk management in software projects: A reassessment. *Journal of Systems and Software,* 81**,** 2118-2133.

BARNEY, S., MOHANKUMAR, V., CHATZIPETROU, P., AURUM, A., WOHLIN, C. & ANGELIS, L. 2014. Software quality across borders: Three case studies on company internal alignment. *Information and Software Technology,* 56**,** 20-38.

BASHAM, M. J. & ROSADO, A. L. 2005. A qualitative analysis of computer security education and training in the United States: An implementation plan for St. Petersburg College. *Journal of Security Education,* 1**,** 81-116.

BASIRATI, M. R., FEMMER, H., EDER, S., FRITZSCHE, M. & WIDERA, A. Understanding changes in use cases: A case study. 2015 IEEE 23rd International Requirements Engineering Conference (RE), 2015. IEEE, 352-361.

BASRI, S. & O'CONNOR, R. V. 2011. A study of knowledge management process practices in very small software companies. *American Journal of Economics and Business Administration,* 3**,** 636.

BASSETT, D., FRY, J. & BALLANTYNE-SCOTT, B. 2010. *Facelifts for Special Libraries: A Practical Guide to Revitalizing Diverse Physical and Digital Spaces*, Elsevier.

BASTEN, D. & SUNYAEV, A. 2014. A systematic mapping of factors affecting accuracy of software development effort estimation. *Communications of the Association for Information Systems,* 34**,** 4.

BAYONA-ORÉ, S., CALVO-MANZANO, J. A., CUEVAS, G. & SAN-FELIU, T. 2014. Critical success factors taxonomy for software process deployment. *Software Quality Journal,* 22**,** 21-48.

BAZAZ, Y., GUPTA, S., PRAKASHRISHI, O. & SHARMA, L. Comparative study of risk assessment models corresponding to risk elements. Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on, 2012. IEEE, 61-66.

BELLAVITE, P. 2003. Complexity science and homeopathy: a synthetic overview. *Homeopathy,* 92**,** 203-212.

BENNETT, L. & NAIR, C. S. 2010. A recipe for effective participation rates for web-based surveys. *Assessment & Evaluation in Higher Education,* 35**,** 357-365.

BERSSANETI, F. T. & CARVALHO, M. M. 2014. Identification of variables that impact project success in Brazilian companies. *International Journal of Project Management*.

BHOOLA, V. 2015. Impact of Project Success Factors in Managing Software Projects in India: An Empirical Analysis. *Business Perspectives and Research,* 3**,** 109-125.

BIEMER, P. P. & LYBERG, L. E. 2003. *Introduction to survey quality*, John Wiley & Sons.

BIRD, C., NAGAPPAN, N., DEVANBU, P., GALL, H. & MURPHY, B. 2009. Does distributed development affect software quality?: an empirical case study of Windows Vista. *Communications of the ACM,* 52**,** 85-93.

BOEHM, B. W. 1991. Software risk management: principles and practices. *Software, IEEE,* 8**,** 32-41.

BONACICH, P. 1972. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology,* 2**,** 113-120.

BOUSSABAINE, A. 2013. *Risk pricing strategies for public-private partnership projects,* John Wiley & Sons.

BOUSSABAINE, H. & VAKILI-ARDEBILI, A. 2010. Topological characteristics of ecological building design complexity. *Intelligent Buildings International,* 2**,** 124-139.

BRANDES, U. 2001. A faster algorithm for betweenness centrality. *Journal of mathematical sociology,* 25**,** 163-177.

BREWER, J. & HUNTER, A. 1989. *Multimethod research: A synthesis of styles*, Sage Publications, Inc.

BRUCKNER, R. M., LIST, B. & SCHIEFER, J. Risk-management for data warehouse systems. International Conference on Data Warehousing and Knowledge Discovery, 2001. Springer, 219-229.

BRYMAN, A. & CRAMER, D. 2011. *Quantitative data analysis with IBM SPSS 17, 18 and 19,* Routledge.

BUDHWAR, P. S. & MATHEW, J. 2007. The relationship of organisational culture with productivity and quality: A study of Indian software organisations. *Employee Relations,* 29**,** 677-695.

BURKE, R. 2013. Project management: planning and control techniques. *New Jersey, USA*.

BURNS, R. B. 1997. *Introduction to research methods*, Addison Wesley Longman.

BUSE, R. P. & ZIMMERMANN, T. Information needs for software development analytics. Proceedings of the 34th international conference on software engineering, 2012. IEEE Press, 987-996.

BUTLER, B., BLAIR, P., FOX, A., HALL, I., HENRY, K. & MCDERMID, J. 2004. The challenges of complex IT projects. London, England: Royal Academy of Engineering. *British Computer Society*.

BUTLER, T. & FITZGERALD, B. 1999. Unpacking the systems development process: an empirical application of the CSF concept in a research context. *The Journal of Strategic Information Systems*, 8**,** 351-371.

CADLE, J. & YEATES, D. 2004. *Project management for information systems*, Pearson education.

CALLEAM 2014. Why Projects Fail: Facts and Figures. *Calleum Consulting*.

CARR, A. S. & SMELTZER, L. R. 2002. The relationship between information technology use and buyer-supplier relationships: an exploratory analysis of the buying firm's perspective. *IEEE Transactions on Engineering Management,* 49**,** 293-304.

CARR, M. J., KONDA, S. L., MONARCH, I., ULRICH, F. C. & WALKER, C. F. 1993. Taxonomy-based risk identification. DTIC Document.

CATALDO, M. & HERBSLEB, J. D. Factors leading to integration failures in global feature-oriented development: an empirical analysis. Proceedings of the 33rd International Conference on Software Engineering, 2011. ACM, 161-170.

CATALDO, M. & NAMBIAR, S. On the relationship between process maturity and geographic distribution: an empirical analysis of their impact on software quality. Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, 2009. ACM, 101-110.

CERPA, N., BARDEEN, M., ASTUDILLO, C. A. & VERNER, J. 2016. Evaluating different families of prediction methods for estimating software project outcomes. *Journal of Systems and Software,* 112**,** 48-64.

CERPA, N. & VERNER, J. M. 2009. Why did your project fail? *Communications of the ACM,* 52**,** 130-134.

CHANDRASEKARAN, S. & KANIYAR, S. 2014. Achieving success in large, complex software projects. *Mckenzie & Company*.

CHARETTE, R. N. 1989. *Software engineering risk analysis and management*, Intertext Publications New York.

CHARETTE, R. N. 2005a. Why software fails. *IEEE spectrum*, 42**,** 36.

CHARETTE, R. N. 2005b. Why software fails [software failure]. *Ieee Spectrum,* 42**,** 42-49.

CHOW, T. & CAO, D.-B. 2008. A survey study of critical success factors in agile software projects. *Journal of Systems and Software*, 81**,** 961-971.

CICMIL, S., WILLIAMS, T., THOMAS, J. & HODGSON, D. 2006. Rethinking project management: researching the actuality of projects. *International Journal of Project Management,* 24**,** 675-686.

CILLIERS, P. & SPURRETT, D. 1999. Complexity and post-modernism: Understanding complex systems. *South African Journal of Philosophy,* 18**,** 258-274.

CLARKE, P. & O'CONNOR, R. V. 2012. The situational factors that affect the software development process: Towards a comprehensive reference framework. *Information and Software Technology,* 54**,** 433-447.

COCKBURN, A. 2006. *Agile software development: the cooperative game*, Pearson Education.

COHEN, D., LINDVALL, M. & COSTA, P. 2004. An introduction to agile methods. *Advances in computers,* 62**,** 1-66.

COLLIS, J. & HUSSEY, R. 2013. *Business research: A practical guide for undergraduate and postgraduate students,* Palgrave macmillan.

COLOMO-PALACIOS, R., CASADO-LUMBRERAS, C., SOTO-ACOSTA, P., GARCÍA-PEÑALVO, F. J. & TOVAR, E. 2014. Project managers in global software development teams: a study of the effects on productivity and performance. *Software Quality Journal,* 22**,** 3-19.

CONSTANTIN, A. & MANURASADA, I. 2015. Customer Collaboration in Software Development: A Systematic Literature Review.

COUTURE, M. & VALCARTIER, D. 2007. Complexity and chaos-state-of-the-art; overview of theoretical concepts. *Minister of National Defence, Canada*.

CRESSWELL, J. W. 1998. Qualitative inquiry and research design: Choosing among five traditions. Sage Publications.

CRESWELL, J. W. 2009. Research design: qualitative, quantitative, and mixed methods approaches. *Chapter One-Selection Of Research Design, 3rd Ed. SAGE publication*.

CRESWELL, J. W. 2013. *Research design: Qualitative, quantitative, and mixed methods approaches*, Sage publications.

CRESWELL, J. W., PLANO CLARK, V. L., GUTMANN, M. L. & HANSON, W. E. 2003. Advanced mixed methods research designs. *Handbook of mixed methods in social and behavioral research*, 209-240.

CROCI, E. & GRASSI, R. 2014. The economic effect of interlocking directorates in Italy: new evidence using centrality measures. *Computational and Mathematical Organization Theory,* 20**,** 89-112.

DADZIE, J. 2005. Understanding software patching. *Queue,* 3**,** 24-30.

DAIM, T. U., KHAMMUANG, M. & GARCES, E. 2016. The Application of Social Network Analysis: Case of Smart Roofing. *Anticipating Future Innovation Pathways Through Large Data Analysis.* Springer.

DARWISH, N. R. & RIZK, N. M. 2015. Multi-Dimensional Success Factors of Agile Software Development Projects. *International Journal of Computer Applications,* 118.

DAVIS, K. 2014. Different stakeholder groups and their perceptions of project success. *International Journal of Project Management,* 32**,** 189-201.

DE BAKKER, K., BOONSTRA, A. & WORTMANN, H. 2010. Does risk management contribute to IT project success? A meta-analysis of empirical evidence. *International Journal of Project Management,* 28**,** 493-503.

DE WIT, A. 1988. Measurement of project success. *International journal of project management,* 6**,** 164-170.

DEMIR, K. A. 2009a. A Survey on Challenges of Software Project Management. *Software Engineering Research and Practice,* 2009**,** 579-585.

DEMIR, K. A. A Survey on Challenges of Software Project Management. Software Engineering Research and Practice, 2009b. 579-585.

DENSCOMBE, M. 2014. *The good research guide: for small-scale social research projects*, McGraw-Hill Education (UK).

DENZIN, N. 2006. Sociological methods: A sourcebook: Aldine Transaction.

DENZIN, N. K. 1970. *The research act in sociology*.

DEUTSKENS, E., DE RUYTER, K., WETZELS, M. & OOSTERVELD, P. 2004. Response rate and response quality of internet-based surveys: An experimental study. *Marketing letters,* 15**,** 21-36.

DEVEDZIC, V. 2007. Software Project Management. *Chang, S., K.(ed.)," Handbook of Software Engineering and Knowledge Engineering,* 2**,** 419-446.

DEY, P. K., KINCH, J. & OGUNLANA, S. O. 2007. Managing risk in software development projects: a case study. *Industrial Management & Data Systems,* 107**,** 284-303.

DICTIONARY, C. 2016a. Cambridge Dictionaries Online [Electronic resource]. Cambridge University Press.

DICTIONARY, O. 2014. Oxford dictionaries. *Language Matters*.

DICTIONARY, O. 2016b. The Oxford Dictionary of New Words: A popular guide to words in the news.

DILLMAN, D. A. 2000. *Mail and internet surveys: The tailored design method*, Wiley New York.

DOROFEE, A. J., WALKER, J. A., ALBERTS, C. J., HIGUERA, R. P. & MURPHY, R. L. 1996. Continuous Risk Management Guidebook. DTIC Document.

DOUGHERTY, R. J. & PAPADOPOULOS, P. E. 2005. Software Risk Identification for Interplanetary Probes.

DU, S., LU, T., ZHAO, L., XU, B., GUO, X. & YANG, H. Towards An Analysis of Software Supply Chain Risk Management. Proceedings of the World Congress on Engineering and Computer Science, 2013.

DURLAND, M. M. & FREDERICKS, K. A. 2005. An introduction to social network analysis. *New Directions for Evaluation,* 2005**,** 5-13.

DYBA, T. 2000. An instrument for measuring the key factors of success in software process improvement. *Empirical software engineering,* 5**,** 357-390.

DYBÅ, T. & DINGSØYR, T. 2008. Empirical studies of agile software development: A systematic review. *Information and software technology,* 50**,** 833-859.

ELETA, I. & GOLBECK, J. 2012. Bridging languages in social networks: How multilingual users of Twitter connect language communities? *Proceedings of the American Society for Information Science and Technology,* 49**,** 1-4.

ELO, S. & KYNGÄS, H. 2008. The qualitative content analysis process. *Journal of advanced nursing,* 62**,** 107-115.

ELZAMLY, A. & HUSSIN, B. Identifying and managing software project risks with proposed fuzzy regression analysis techniques: Maintenance phase. Conference on Management and Engineering (CME2014), 2014. 1868-1881.

ELZAMLY, A., HUSSIN, B. & SALLEH, N. M. 2016. Top Fifty Software Risk Factors and the Best Thirty Risk Management Techniques in Software Development Lifecycle for Successful Software Projects. *International Journal of Hybrid Information Technology,* 9**,** 11-32.

ERIKSSON, P. & KOVALAINEN, A. 2015. *Qualitative Methods in Business Research: A Practical Guide to Social Research*, Sage.

ESTRADA, E., FOX, M., HIGHAM, D. J. & OPPO, G.-L. 2010. *Network Science: Complexity in Nature and Technology*, Springer Science & Business Media.

EVALUATION, U. S. G. A. O. P. & DIVISION, M. 1996. *Content analysis: A methodology for structuring and analyzing written material*, US General Accounting Office.

EVERETT, M. & BORGATTI, S. P. 2005. Ego network betweenness. *Social networks,* 27**,** 31-38.

FANG, C. & MARLE, F. 2013. Dealing with project complexity by matrix-based propagation modelling for project risk analysis. *Journal of Engineering Design,* 24**,** 239-256.

FANG, C., MARLE, F., ZIO, E. & BOCQUET, J.-C. 2012. Network theory-based analysis of risk interactions in large engineering projects. *Reliability Engineering & System Safety,* 106**,** 1-10.

FITSILIS, P. Measuring the Complexity of Software Projects. CSIE (7), 2009. 644-648.

FONTANA, M. 2010. Can neoclassical economics handle complexity? The fallacy of the oil spot dynamic. *Journal of Economic Behavior & Organization,* 76**,** 584-596.

FOWLER JR, F. J. 2013. *Survey research methods*, Sage publications.

FRAME, J. D. 2002. *The new project management: tools for an age of rapid change, complexity, and other business realities*, John Wiley & Sons.

FRASER, G. & ARCURI, A. 2014. A large-scale evaluation of automated unit test generation using EvoSuite. *ACM Transactions on Software Engineering and Methodology (TOSEM),* 24**,** 8.

FREEMAN, L. C. 1978. Centrality in social networks conceptual clarification. *Social networks,* 1**,** 215-239.

FRUDE, N. 1987. *A Guide to Spss/Pc+*, Springer.

GALLAGHER, B. P., CASE, P. J., CREEL, R. C., KUSHNER, S. & WILLIAMS, R. C. 2005. A taxonomy of operational risks.

GANDHI, A., NAIK, A., THAKKAR, K. & GAHIRWAL, M. 2014. Risk Management in Software Development using Artificial Neural Networks. *International Journal of Computer Applications,* 93.

GANJI, D., MOURATIDIS, H., GHEYTASSI, S. M. & PETRIDIS, M. 2015. Conflicts Between Security and Privacy Measures in Software Requirements Engineering. *Global Security, Safety and Sustainability: Tomorrow's Challenges of Cyber Security.* Springer.

GATRELL, A. C. 2005. Complexity theory and geographies of health: a critical assessment. *Social Science & Medicine,* 60**,** 2661-2671.

GEA, D., CARRILLO, J. M., NICOLAS, J.-M., ALEMAN, J., TOVAL, A., EBERT, C. & VIZCAINO, A. 2011. Requirements engineering tools. *Software, IEEE,* 28**,** 86-91.

GIBBS, G. 2002. *Qualitative data analysis: Explorations with NVivo (Understanding social research),* Buckingham: Open University Press.

GROUP, S. 2013. Chaos manifesto. *The Standish Group*.

GROVES, R. M., FOWLER JR, F. J., COUPER, M. P., LEPKOWSKI, J. M., SINGER, E. & TOURANGEAU, R. 2009. *Survey methodology*, John Wiley & Sons.

GUBA, E. G. & LINCOLN, Y. S. 1994. Competing paradigms in qualitative research. *Handbook of qualitative research,* 2**,** 105.

GUO, Y. & SEAMAN, C. B. A survey of software project managers on software process change. Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, 2008. ACM, 263-269.

HADAD, Y., KEREN, B. & LASLO, Z. 2013. A decision-making support system module for project manager selection according to past performance. *International Journal of Project Management,* 31**,** 532-541.

HAN, W.-M. & HUANG, S.-J. 2007. An empirical analysis of risk components and performance on software projects. *Journal of Systems and Software,* 80**,** 42-50.

HASHIMI, H., HAFEZ, A. & BERAKA, M. A novel view of risk management in software development life cycle. Pervasive Systems, Algorithms and Networks (ISPAN), 2012 12th International Symposium on, 2012. IEEE, 128-134.

HASTIE, S. & WOJEWODA, S. 2015a. Standish Group 2015 Chaos Report. Londres: Chaos.

HASTIE, S. & WOJEWODA, S. 2015b. Standish Group 2015 Chaos Report-Q&A with Jennifer Lynch. *Retrieved,* 1**,** 2016.

HE, J., BUTLER, B. S. & KING, W. R. 2007. Team cognition: Development and evolution in software project teams. *Journal of Management Information Systems,* 24**,** 261-292.

HEALY, M. & PERRY, C. 2000. Comprehensive criteria to judge validity and reliability of qualitative research within the realism paradigm. *Qualitative market research: An international journal,* 3**,** 118-126.

HEYMANN, S. & LE GRAND, B. Visual analysis of complex networks for business intelligence with gephi. 1st International Symposium on Visualisation and Business Intelligence, in conjunction with the 17th International Conference Information Visualisation, 2013.

HIGUERA, R. P. & HAIMES, Y. Y. 1996. Software Risk Management. DTIC Document.

HIJAZI, H., ALQRAINY, S., MUAIDI, H. & KHDOUR, T. 2014a. Identifying Causality Relation between Software Projects Risk Factors. *International Journal of Software Engineering and Its Applications,* 8**,** 51-58.

HIJAZI, H., ALQRAINY, S., MUAIDI, H. & KHDOUR, T. 2014b. Risk factors in software development phases. *European Scientific Journal,* 10.

HIJAZI, H., KHDOUR, T. & ALARABEYYAT, A. 2012. A review of risk management in different software development methodologies. *International Journal of Computer Applications,* 45**,** 8-12.

HILL, J. A software safety risk taxonomy for use in retrospective safety cases. Software Engineering Workshop, 2007. SEW 2007. 31st IEEE, 2007. IEEE, 179-186.

HILL, T. 1991. *Production/operations management: text and cases,* Prentice Hall.

HOERMANN, S., SCHERMANN, M. & KRCMAR, H. 2011. Variations in Risk Exposure: A Longitudinal Analysis of Enterprise Software Projects. *VARIATIONS,* 10**,** 6-2011.

HOFSTEDE, G., NEUIJEN, B., OHAYV, D. D. & SANDERS, G. 1990. Measuring organizational cultures: A qualitative and quantitative study across twenty cases. *Administrative science quarterly***,** 286-316.

HOLES, S. 1992. The social structure of competition. Cambridge, MA: Harvard University Press.

HOODAT, H. & RASHIDI, H. 2009. Classification and analysis of risks in software engineering. *World Academy of Science, Engineering and Technology,* 56**,** 446-452.

HOSKINS, C. N. & MARIANO, C. 2004. *Research in nursing and health: Understanding and using quantitative and qualitative methods*, Springer Publishing Company.

HOSSAIN, S. & BROOKS, L. 2008. Fuzzy cognitive map modelling educational software adoption. *Computers & Education,* 51**,** 1569-1588.

HOX, J. J. & BOEIJE, H. R. 2005. Data collection, primary vs. secondary. *Encyclopedia of social measurement,* 1**,** 593-599.

HU, Y., ZHANG, X., NGAI, E., CAI, R. & LIU, M. 2013. Software project risk analysis using Bayesian networks with causality constraints. *Decision Support Systems,* 56**,** 439-449.

HUIJGENS, H., VAN DEURSEN, A. & VAN SOLINGEN, R. Success factors in managing legacy system evolution: a case study. Proceedings of the International Workshop on Software and Systems Process, 2016. ACM, 96-105.

HUNG, S.-Y., YU, W.-J., CHEN, C. C. & HSU, J.-C. 2013. Enhancing Organizational Fit Factors to Achieve ERP Project Success. *Pacific Asia Journal of the Association for Information Systems,* 5.

HUO, M., VERNER, J., ZHU, L. & BABAR, M. A. Software quality and agile methods. Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International, 2004. IEEE, 520-525.

HUSSAIN, A., MKPOJIOGU, E. O. & KAMAL, F. M. 2016. The Role of Requirements in the Success or Failure of Software Projects. *International Review of Management and Marketing,* 6.

HYMAN, M. R. & YANG, Z. 2001. International marketing serials: a retrospective. *International Marketing Review,* 18**,** 667-718.

IKA, L. A. 2009. Project success as a topic in project management journals. *Project Management Journal*, 40**,** 6-19.

IKONEN, M. & ABRAHAMSSON, P. 2013. Operationalizing the concept of success in software engineering projects. *Business Innovation, Development, and Advancement in the Digital Economy***,** 89.

IMTIAZ, M. A., AL-MUDHARY, A. S., MIRHASHEMI, M. T. & IBRAHIM, R. 2013. Critical Success Factors of Information Technology Projects. *World Academy of Science, Engineering and Technology, International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering,* 7**,** 3154-3158.

IQBAL, M. A., AZAM, F. & QURESHI, A. A. 2011. Critical aspects of project management causing major impacts on software development. *International Journal of Computer Applications (0975–8887)*.

IRANI, Z., SHARIF, A. M. & LOVE, P. E. 2001. Transforming failure into success through organisational learning: an analysis of a manufacturing information system. *European Journal of Information Systems,* 10**,** 55-66.

IRANMANESH, S. H., KHODADADI, S. B. & TAHERI, S. Risk assessment of software projects using fuzzy inference system. Computers & Industrial Engineering, 2009. CIE 2009. International Conference on, 2009. IEEE, 1149-1154.

ISLAM, S., JOARDER, M. M. A. & HOUMB, S. H. Goal and risk factors in offshore outsourced software development from vendor's viewpoint. 2009 Fourth IEEE International Conference on Global Software Engineering, 2009. IEEE, 347-352.

ISLAM, S., MOURATIDIS, H. & WEIPPL, E. R. 2014. An empirical study on the implementation and evaluation of a goal-driven software development risk management model. *Information and Software Technology,* 56**,** 117-133.

ISO, I. 2009. 31000: 2009 Risk management–Principles and guidelines. *International Organization for Standardization, Geneva, Switzerland*.

JAGER, C.-J., VOS, S., BORGERS, M., HARMSEN, F., BRINKKEMPER, S. & WIJNGAERT, L. Controlling risk prior to offshore application development. Proceedings Global Sourcing Workshop, Val d'Isère, 2008.

JALIL, Z. Is Non Technical Person a Better Software Project Manager? Computer Science and Software Engineering, 2008 International Conference on, 2008. IEEE, 1-5.

JAMIESON, K. & HYLAND, P. 2006. Factors that influence Information Systems decisions and outcomes: A summary of key themes from four case studies. *ACIS 2006 Proceedings***,** 43.

JIANG, J. J., KLEIN, G., HWANG, H.-G., HUANG, J. & HUNG, S.-Y. 2004. An exploration of the relationship between software development process maturity and project performance. *Information & Management,* 41**,** 279-288.

JIANGPING, W., LIANYU, L. & DAN, W. 2013. Research on Risk Factors of ICT Commercialization with Grounded Theory. *Journal of Computers,* 8**,** 2356-2365.

JOGULU, U. D. & PANSIRI, J. 2011. Mixed methods: A research design for management doctoral dissertations. *Management research review,* 34**,** 687-701.

JOHNSON, N. 2009. *Simply Complexity: A clear guide to complexity theory*, Oneworld Publications.

JØRGENSEN, M. 2014. Failure factors of small software projects at a global outsourcing marketplace. *Journal of Systems and Software,* 92**,** 157-169.

JUÁREZ-RAMÍREZ, R., PIMIENTA-ROMO, R. & OCEGUEDA-MIRAMONTES, V. Using social networks for integrating a tacit knowledge repository to support the performance of software development teams. International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making, 2013. Springer, 167-179.

KAMUNI, S. K. 2015. Study of Factors that Induce Software Project Overrun Time.

KAUR, K., JYOTI, B. & RANI, R. 2013. Analysis of gold plating: a software development risk. *International Journal of Computer Science and Communication Engineering*.

KAY, J. J. 2004. On complexity theory, exergy and industrial ecology. *Construction Ecology: Nature as the Basis for Green Buildings; Kibert, C., Sendzimir, J., Guy, B., Eds***,** 72-107.

KEESANG, L., SUNGHO, K. & SAKAWA, M. 1996. on-line Fault Diagnosis by Using Fuzzy Cognitive Map. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences,* 79**,** 921-927.

KEIL, M., CULE, P. E., LYYTINEN, K. & SCHMIDT, R. C. 1998. A framework for identifying software project risks. *Communications of the ACM,* 41**,** 76-83.

KEIL, M., LEE, H. K. & DENG, T. 2013. Understanding the most critical skills for managing IT projects: A Delphi study of IT project managers. *Information & Management,* 50**,** 398-414.

KEIL, M., LI, L., MATHIASSEN, L. & ZHENG, G. 2008. The influence of checklists and roles on software practitioner risk perception and decision-making. *Journal of Systems and Software,* 81**,** 908-919.

KEIL, M., RAI, A., MANN, J. C. & ZHANG, G. P. 2003. Why software projects escalate: The importance of project management constructs. *IEEE Transactions on Engineering Management,* 50**,** 251-261.

KELLEY, K., CLARK, B., BROWN, V. & SITZIA, J. 2003. Good practice in the conduct and reporting of survey research. *International Journal for Quality in Health Care,* 15**,** 261-266.

KHAN, A. W. & KHAN, S. U. 2013. Critical success factors for offshore software outsourcing contract management from vendors' perspective: an exploratory study using a systematic literature review. *IET software,* 7**,** 327-338.

KHAN, S. U., NIAZI, M. & AHMAD, R. 2011. Factors influencing clients in the selection of offshore software outsourcing vendors: An exploratory study using a systematic literature review. *Journal of systems and software,* 84**,** 686-699.

KHATAVAKHOTAN, A. S. & OW, S. H. Improving IT Risk Management Process by an Embedded Dynamic Verifier Core: Towards Reducing IT Projects Failure. Intelligent Systems, Modelling and Simulation (ISMS), 2012 Third International Conference on, 2012. IEEE, 684-687.

KIBE, A. M. 2015. *Risk Management of Financial Information Systems Using Bayesian Networks.*

KIM, H.-W. & KANKANHALLI, A. 2009. Investigating user resistance to information systems implementation: A status quo bias perspective. *Mis Quarterly***,** 567-582.

KIRK, D. & MACDONELL, S. 2009. A systems approach to software process improvement in small organisations.

KOH, A. 2011. The roles, responsibilities and practices of project portfolio manager in Australia: A qualitative study.

KOMBO, D. K. & TROMP, D. L. 2006. Proposal and thesis writing: An introduction. *Nairobi: Paulines Publications Africa***,** 10-45.

KONTIO, J., GETTO, G. & LANDES, D. Experiences in improving risk management processes using the concepts of the Riskit method. ACM SIGSOFT Software Engineering Notes, 1998. ACM, 163-174.

KOOPMAN, P. Risk areas in embedded software industry projects. Proceedings of the 2010 Workshop on Embedded Systems Education, 2010. ACM, 5.

KOOPMAN, P. 2011. Avoiding the Top 43 Embedded Software Risks.

KOSKO, B. 1986. Fuzzy cognitive maps. *International Journal of man-machine studies,* 24**,** 65-75.

KUMARI, N. & PILLAI, A. S. A study on project scope as a requirements elicitation issue. Computing for Sustainable Global Development (INDIACom), 2014 International Conference on, 2014. IEEE, 510-514.

LAGERSTROM, R., BALDWIN, C., MACCORMACK, A. & AIER, S. Visualizing and measuring enterprise application architecture: an exploratory telecom case. System Sciences (HICSS), 2014 47th Hawaii International Conference on, 2014. IEEE, 3847-3856.

LANGER, N., SLAUGHTER, S. A. & MUKHOPADHYAY, T. 2014. Project managers' practical intelligence and project performance in software offshore outsourcing: A field study. *Information Systems Research,* 25**,** 364-384.

LAPOINTE, L. & RIVARD, S. 2005. A multilevel model of resistance to information technology implementation. *MIS quarterly***,** 461-491.

LAURI, S. & KYNGAS, H. 2005. Developing nursing theories. *Werner Söderström, Dark Oy: Vantaa, Finland*.

LEE, B.-C. & RHEW, S. Y. 2014. An Empirical Study on Adjustment Factors to Estimate Maintenance Cost of Applications Developed Using Components. *Lecture Notes on Software Engineering,* 2**,** 90.

LEE, G. & XIA, W. 2010. Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *Mis Quarterly,* 34**,** 87-114.

LEE, J. S., KEIL, M. & KASI, V. 2012. The effect of an initial budget and schedule goal on software project escalation. *Journal of Management Information Systems,* 29**,** 53-78.

LEE, S.-K. & YU, J.-H. 2012. Success model of project management information system in construction. *Automation in construction,* 25**,** 82-93.

LEHTINEN, T. O., MÄNTYLÄ, M. V., VANHANEN, J., ITKONEN, J. & LASSENIUS, C. 2014. Perceived causes of software project failures–An analysis of their relationships. *Information and Software Technology,* 56**,** 623-643.

LEVESQUE, L. L., WILSON, J. M. & WHOLEY, D. R. 2001. Cognitive divergence and shared mental models in software development project teams. *Journal of Organizational Behavior,* 22**,** 135-144.

LI, J., CONRADI, R., SLYNGSTAD, O. P., TORCHIANO, M., MORISIO, M. & BUNSE, C. 2008. A state-of-the-practice survey of risk management in development with off-the-shelf software components. *IEEE Transactions on Software Engineering,* 34**,** 271-286.

LI, J., LI, M., WU, D. & SONG, H. 2012. An integrated risk measurement and optimization model for trustworthy software process management. *Information Sciences,* 191**,** 47-60.

LI, Q. 2013. A novel Likert scale based on fuzzy sets theory. *Expert Systems with Applications,* 40**,** 1609-1618.

LI, Y. 2016. Research Design and Methodology. *Expatriate Manager's Adaption and Knowledge Acquisition.* Springer.

LIANG, T.-P., LIU, C.-C., LIN, T.-M. & LIN, B. 2007. Effect of team diversity on software project performance. *Industrial Management & Data Systems,* 107**,** 636-653.

LIMON, S., YADAV, O. P. & NEPAL, B. Modeling cognitive network of a physical system using design knowledge base. 2014 IEEE International Conference on Industrial Engineering and Engineering Management, 2014. IEEE, 238-242.

LINBERG, K. R. 1999. Software developer perceptions about software project failure: a case study. *Journal of Systems and Software,* 49**,** 177-192.

LIU, S., ZHANG, J., KEIL, M. & CHEN, T. 2010. Comparing senior executive and project manager perceptions of IT project risk: a Chinese Delphi study. *Information Systems Journal,* 20**,** 319-355.

LOVE, P. E., IRANI, Z., STANDING, C., LIN, C. & BURN, J. M. 2005. The enigma of evaluation: benefits, costs and risks of IT in Australian small–medium-sized enterprises. *Information & Management,* 42**,** 947-964.

LU, S.-T. & YU, S.-H. 2012. Risk Factors Assessment for Software DevelopmentProject Based on Fuzzy Decision Making. *International Journal of Information and Electronics Engineering,* 2**,** 596.

LU, S.-T., YU, S.-H., CHANG, D.-S. & SU, S.-C. 2013. Using the Fuzzy Linguistic preference relation approach for assessing the importance of risk factors in a software development project. *Mathematical Problems in Engineering*, 2013.

LU, X., LIU, H. & YE, W. Analysis failure factors for small & medium software projects based on PLS method. Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on, 2010. IEEE, 676-680.

LU, Y., LUO, L., WANG, H., LE, Y. & SHI, Q. 2015. Measurement model of project complexity for large-scale projects from task and organization perspective. *International Journal of Project Management*, 33**,** 610-622.

MARSAVINA, C. 2014. *Studying fine-grained co-evolution patterns of production and test code.* TU Delft, Delft University of Technology.

MARUPING, L. M., ZHANG, X. & VENKATESH, V. 2009. Role of collective ownership and coding standards in coordinating expertise in software project teams. *European Journal of Information Systems,* 18**,** 355-371.

MATHISON, S. 1988. Why triangulate? *Educational researcher,* 17**,** 13-17.

MAYLOR, H. & BLACKMON, K. 2005. *Researching business and management: A roadmap for success*, Palgrave Macmillan.

MCCRUM-GARDNER, E. 2008. Which is the correct statistical test to use? *British Journal of Oral and Maxillofacial Surgery,* 46**,** 38-41.

MCLEOD, L. & MACDONELL, S. G. 2011. Factors that affect software systems development project outcomes: A survey of research. *ACM Computing Surveys (CSUR),* 43**,** 24.

MCMANUS, J. 2012. *Risk management in software development projects*, Routledge.

MCNEILL, P. 1990. Society Now Research Methods. New York: Routledge.

MCSWEENEY, P. J., MEHROTRA, K. & OH, J. C. 2014. Game-theoretic framework for community detection. *Encyclopedia of Social Network Analysis and Mining.* Springer.

MENDES, E. A comparison of techniques for web effort estimation. Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on, 2007. IEEE, 334-343.

MENEZES JR, J., GUSMÃO, C. & MOURA, H. 2013. Defining Indicators for Risk Assessment in Software Development Projects. *Clei electronic journal,* 16**,** 11-11.

MIKULECKY, D. C. 2001. The emergence of complexity: science coming of age or science growing old? *Computers & chemistry,* 25**,** 341-348.

MIORANDO, R. F., RIBEIRO, J. L. D. & CORTIMIGLIA, M. N. 2014. An economic–probabilistic model for risk analysis in technological innovation projects. *Technovation,* 34**,** 485-498.

MISHRA, D. & MISHRA, A. 2011. A review of non-technical issues in global software development. *International Journal of Computer Applications in Technology,* 40**,** 216-224.

MITCHELL, M. 2009. *Complexity: A guided tour*, Oxford University Press.

MOORTHY, J. T. S. 2014. Identification of usability risk as a risk in software development projects. *International Journal of Software Engineering and Technology,* 1.

MOORTHY, J. T. S., BIN IBRAHIM, S. & MAHRIN, M. N. R. Formulation of Usability Risk Assessment Model. Open Systems (ICOS), 2013 IEEE Conference on, 2013. IEEE, 168-173.

MORRISON, A., MCMILLAN, D., REEVES, S., SHERWOOD, S. & CHALMERS, M. A hybrid mass participation approach to mobile software trials. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2012. ACM, 1311-1320.

MORSE, J., TASHAKKORI, A. & TEDDLIE, C. 2003. Handbook of mixed methods in social and behavioral research. *Handbook of mixed methods in social & behavioral research*.

MURPHY, B., BIRD, C., ZIMMERMANN, T., WILLIAMS, L., NAGAPPAN, N. & BEGEL, A. Have agile techniques been the silver bullet for software development at microsoft? 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2013. IEEE, 75-84.

MURTHI, S. 2002. Preventive risk management software for software projects. *IT professional,* 4**,** 9-15.

NA, K.-S., SIMPSON, J. T., LI, X., SINGH, T. & KIM, K.-Y. 2007. Software development risk and project performance measurement: Evidence in Korea. *Journal of Systems and Software,* 80**,** 596-605.

NAIR, C. S. & ADAMS, P. 2009. Survey platform: A factor influencing online survey delivery and response rate. *Quality in Higher Education,* 15**,** 291-296.

NAKATSU, R. T. & IACOVOU, C. L. 2009. A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel Delphi study. *Information & Management,* 46**,** 57-68.

NANDHAKUMAR, J. 1996. Design for success?: critical success factors in executive information systems development. *European Journal of Information Systems,* 5**,** 62-72.

NASIR, M. & SAHIBUDDIN, S. 2011a. Addressing a critical success factor for software projects: A multi-round Delphi study of TSP. *International journal of the physical sciences,* 6**,** 1213-1232.

NASIR, M. H. N. & SAHIBUDDIN, S. 2011b. Critical success factors for software projects: A comparative study. *Scientific research and essays,* 6**,** 2174-2186.

NEVILLE, C. 2007. Introduction to research and research methods. *Bradford: Effective Learning Service*.

NEWTON, P. 2015. Managing the Project Budget.

NGUYEN-DUC, A., CRUZES, D. S. & CONRADI, R. 2015. The impact of global dispersion on coordination, team performance and software quality–A systematic literature review. *Information and Software Technology,* 57**,** 277-294.

NGUYEN, T. N. Software project management towards failure avoidance. Software Engineering and Applications (ICSOFT-EA), 2014 9th International Conference on, 2014. IEEE, 560-567.

NIAZI, M. & BABAR, M. A. 2009. Identifying high perceived value practices of CMMI level 2: an empirical study. *Information and software technology,* 51**,** 1231-1243.

NIDHRA, S., YANAMADALA, M., AFZAL, W. & TORKAR, R. 2013. Knowledge transfer challenges and mitigation strategies in global software development—A systematic literature review and industrial validation. *International journal of information management,* 33**,** 333-355.

NIENABER, A.-M., SCHEWE, G., HOLTORF, V. & LEKER, J. Initiative in teams in the front end of innovation: An empirical study about the success of structured and unstructured teams. 2012 Proceedings of PICMET'12: Technology Management for Emerging Technologies, 2012. IEEE, 2437-2449.

O'CONNOR, R. V. & LAPORTE, C. Y. 2012. *Software project management in very small entities with ISO/IEC 29110,* Springer.

OLAZABAL, M. & RECKIEN, D. 2015. 7 Fuzzy cognitive mapping: applications to urban environmental decision-making. *Handbook of Research methods and Applications in Environmental Studies***,** 148.

ORTEGA, J. L. 2014. Influence of co-authorship networks in the research impact: Ego network analyses from Microsoft Academic Search. *Journal of Informetrics,* 8**,** 728-737.

OTERO, L. D., CENTENO, G., RUIZ-TORRES, A. J. & OTERO, C. E. 2009. A systematic approach for resource allocation in software projects. *Computers & Industrial Engineering,* 56**,** 1333-1339.

OTEYO, I. N., NAMUSONGE, G. S. & MUTUA, J. 2014. The Dynamics of Implementing Software Engineering Projects. *International Journal of Computer (IJC),* 12**,** 23-32.

PALM, K. & LINDAHL, M. 2015. A project as a workplace: Observations from project managers in four R&D and project-intensive companies. *International Journal of Project Management,* 33**,** 828-838.

PAPAGEORGIOU, E. I. 2013. *Fuzzy Cognitive Maps for Applied Sciences and Engineering: From Fundamentals to Extensions and Learning Algorithms*, Springer Science & Business Media.

PAPAKOSTAS, G., PAPAGEORGIOU, E. & KABURLASOS, V. Linguistic Fuzzy Cognitive Map (LFCM) for pattern recognition. Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on, 2015. IEEE, 1-7.

PARKER, D. & STACEY, R. 1994. *Chaos, management and economics: The implications of nonlinear thinking*, Institute of Economic Affairs.

PARRY, S., KUPIEC-TEAHAN, B. & ROWLEY, J. 2011. Exploring marketing and relationships in software SMEs: A mixed methods approach. *Management Research Review,* 35**,** 52-68.

PEREIRA, J., CERPA, N., VERNER, J., RIVAS, M. & PROCACCINO, J. D. 2008. What do software practitioners really think about project success: A cross-cultural comparison. *Journal of Systems and Software,* 81**,** 897-907.

PERERA, M. & RANASINGHE, M. Prompt list for risk management in Sri Lankan software industry. Information and Automation, 2006. ICIA 2006. International Conference on, 2006. IEEE, 7-12.

PETKOVA, O. & PETKOV, D. Towards A Better Understanding Of Factors Affecting Software Project Complexity. Proceedings of the Conference on Information Systems Applied Research ISSN, 2015. 1508.

PHILGE, N. 2014. Agile Software Development Methodology: A case study of challenges and issues associated with applying Scrum.

PHUA, F. T. 2004. Modelling the determinants of multi-firm project success: a grounded exploration of differing participant perspectives. *Construction Management and Economics,* 22**,** 451-459.

PLATT, H., PLATT, H. & RICHARDS, J. 1992. Dictionary of language teaching and applied linguistics. *Essex: Longman*.

POTDA, M. S. M., INGLE, A. & PURI, M. FACTORS INFLUENCING ON COST ESTIMATION FOR SOFTWARE DEVELOPMENT.

POWELL, P. L. & KLEIN, J. H. 1996. Risk management for information systems development. *Journal of Information Technology,* 11**,** 309-319.

PRASAD, A., ARSIWALA, J. & SINGH, P. P. Estimating and improving the probability of success of a software project by analysing the factors involved using data mining. Artificial Intelligence and Education (ICAIE), 2010 International Conference on, 2010. IEEE, 391-394.

PRESSMAN, R. S. 2010. Software Engineering: A Practitioner's Approach, 7/e, RS Pressman & Associates. *Inc., McGraw-Hill, ISBN,* 73375977.

PUHEIM, M., VAŠČÁK, J. & MADARÁSZ, L. A proposal for Multi-Purpose Fuzzy Cognitive Maps library for complex system modeling. Applied Machine Intelligence and Informatics (SAMI), 2015 IEEE 13th International Symposium on, 2015. IEEE, 175-180.

PURNA SUDHAKAR, G. 2012. A model of critical success factors for software projects. *Journal of Enterprise Information Management,* 25**,** 537-558.

QUADRI, A. T., KOMAL, M. & KHALIL, Z. 2015. A Comprehensive Study on Risk Analysis and Risk Management in IT Industry.

RAITH, F., RICHTER, I., LINDERMEIER, R. & KLINKER, G. Identification of inaccurate effort estimates in agile software development. 2013 20th Asia-Pacific Software Engineering Conference (APSEC), 2013. IEEE, 67-72.

RAMASUBBU, N., CATALDO, M., BALAN, R. K. & HERBSLEB, J. D. Configuring global software teams: a multi-company analysis of project productivity, quality, and profits. Proceedings of the 33rd International Conference on Software Engineering, 2011. ACM, 261-270.

RANE, P. B., PATIL, S., KULKARNI, P. & MESHRAM, B. 2011. Tying Web Project Estimation with Risk Management. *IJCEM International Journal of Computational Engineering & Management,* 14.

REEL, J. S. 1999. Critical success factors in software projects. *Software, IEEE,* 16**,** 18-23.

REIFER, D. 2002. Ten deadly risks in internet and intranet software development. *IEEE software,* 19**,** 12-14.

RINGIM, K. J. & YUSSOF, I. M. 2014. ASSESSING THE INSIGHT OF MUSLIM ACCOUNT HOLDERS'IN CONVENTIONAL BANK TO PURCHASE ISLAMIC BANK PRODUCTS. *Journal of Emerging Economies and Islamic Research (JEEIR),* 2.

RODRIGUES, A. & BOWERS, J. 1996. The role of system dynamics in project management. *International Journal of Project Management,* 14**,** 213-220.

ROSENCRANCE, L. 2007. Survey: Poor communication causes most IT project failures. *Computerworld*.

RUAN, D. 2010. Choquet integral based aggregation approach to software development risk assessment. *Information Sciences,* 180**,** 441-451.

RUHE, G. & WOHLIN, C. 2014a. *Software Project Management in a Changing World*, Springer.

RUHE, G. & WOHLIN, C. 2014b. Software Project Management: Setting the Context. *Software Project Management in a Changing World.* Springer.

RYAN, S. & O'CONNOR, R. V. 2009. Development of a team measure for tacit knowledge in software development teams. *Journal of Systems and Software,* 82**,** 229-240.

SAFA'A, I. H. 2012. Critical Risk Factors for Information System (IS) Projects. Citeseer.

SAINI, S. & HOODA, S. 2014. Critical Aspects of Project Management Causing Major Impacts on Software Development. *International Journal of Research,* 1**,** 1894-1902.

SAJAD, M., SADIQ, M., NAVEED, K. & IQBAL, M. S. 2016. Software Project Management: Tools assessment, Comparison and suggestions for future development. *International Journal of Computer Science and Network Security (IJCSNS),* 16**,** 31.

SALE, J. E., LOHFELD, L. H. & BRAZIL, K. 2002. Revisiting the quantitative-qualitative debate: Implications for mixed-methods research. *Quality and quantity,* 36**,** 43-53.

SALTZ, J. S. The need for new processes, methodologies and tools to support big data teams and improve big data project effectiveness. Big Data (Big Data), 2015 IEEE International Conference on, 2015. IEEE, 2066-2071.

SAMANTRA, C., DATTA, S. & MAHAPATRA, S. S. 2014. Risk assessment in IT outsourcing using fuzzy decision-making approach: An Indian perspective. *Expert Systems with Applications,* 41**,** 4010-4022.

SANTOS, M. D. A., BERMEJO, P. H. D. S., DE OLIVEIRA, M. S., TONELLI, A. O. & SEIDEL, E. J. 2013. Improving the management of cost and scope in software projects using agile practices. *arXiv preprint arXiv:1303.1971*.

SARIGIANNIDIS, L. & CHATZOGLOU, P. D. 2013. Quality vs risk: An investigation of their relationship in software development projects. *International Journal of Project Management*.

SARIGIANNIDIS, L. & CHATZOGLOU, P. D. 2014. Quality vs risk: An investigation of their relationship in software development projects. *International Journal of Project Management,* 32**,** 1073-1082.

SAUER, C. & CUTHBERTSON, C. 2003. The state of IT project management in the UK 2002-2003.

SAUNDERS, M., LEWIS, P. & THORNHILL, A. 2009. Research Methods for Business Students. Harlow, England: Pearson Education. ISBN 978-0-273-71686-0.

SAURIN, T. A., ROOKE, J. & KOSKELA, L. 2013. A complex systems theory perspective of lean production. *International Journal of Production Research,* 51**,** 5824-5838.

SAVELSBERGH, C. M., HAVERMANS, L. A. & STORM, P. 2016. Development paths of project managers: What and how do project managers learn from their experiences? *International Journal of Project Management,* 34**,** 559-569.

SCHMIDT, R., LYYTINEN, K., KEIL, M. & CULE, P. 2001a. Identifying software project risks: an international Delphi study. *Journal of Management Information Systems,* 17**,** 5-36.

SCHMIDT, R., LYYTINEN, K. & MARK KEIL, P. C. 2001b. Identifying software project risks: An international Delphi study. *Journal of management information systems,* 17**,** 5-36.

SCHOTTNER, M. 2003. Algorithms for the application of Hartmann-Shack wavefront sensors in ophthalmology.

SCHÜLER, K., TROGUS, R., FEILKAS, M. & KINNEN, T. Managing Product Quality in Complex Software Development Projects. Proceedings of the Embedded World Conference, 2015.

SELF, R. J. & AQUILINA, C. TechnoStress in the 21st Century; Does It Still Exist and How Does It Affect Knowledge Management and Other Information Systems Initiatives. 7th International Conference on Knowledge Management in Organizations: Service and Cloud Computing, 2013. Springer, 117-127.

SERRADOR, P. 2014. *Project Planning and Project Success: The 25% Solution*, CRC Press.

SERRADOR, P. & TURNER, R. 2015. What is Enough Planning? Results From a Global Quantitative Study. *Engineering Management, IEEE Transactions on,* 62**,** 462-474.

SHAHZAD, B. & AL-MUDIMIGH, A. S. Risk Identification, Mitigation and Avoidance Model for Handling Software Risk. Computational Intelligence, Communication Systems and Networks (CICSyN), 2010 Second International Conference on, 2010. IEEE, 191-196.

SHAHZAD, B. & IQBAL, J. Software Risk Management–Prioritization of frequently occurring Risk in Software Development Phases. Using Relative Impact Risk Model‖. 2nd International Conference on Information and Communication Technology(ICICT2007), 2007.

SHEFFIELD, J. & LEMÉTAYER, J. 2013. Factors associated with the software development agility of successful projects. *International Journal of Project Management,* 31**,** 459-472.

SHEPERIS, C. J., YOUNG, J. S. & DANIELS, M. H. 2016. *Counseling research: Quantitative, qualitative, and mixed methods*, Pearson.

SILA, I. & EBRAHIMPOUR, M. 2002. An investigation of the total quality management survey based research published between 1989 and 2000: A literature review. *International Journal of Quality & Reliability Management,* 19**,** 902-970.

SILVA, F. S., SOARES, F. S. F., PERES, A. L., DE AZEVEDO, I. M., VASCONCELOS, A. P. L., KAMEI, F. K. & DE LEMOS MEIRA, S. R. 2015. Using CMMI together with agile software development: A systematic review. *Information and Software Technology,* 58**,** 20-43.

SIMON, H. A. 1991. The architecture of complexity. *Facets of systems science.* Springer.

SIPAYUNG, J. J. P. & SEMBIRING, J. Risk assessment model of application development using Bayesian Network and Boehm's Software Risk Principles. 2015 International Conference on Information Technology Systems and Innovation (ICITSI), 2015. IEEE, 1-5.

ŠMITE, D. 2006. Global software development projects in one of the biggest companies in Latvia: is geographical distribution a problem? *Software Process: Improvement and Practice,* 11**,** 61-76.

SONCHAN, P. & RAMINGWONG, S. Top twenty risks in software projects: A content analysis and Delphi study. Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2014 11th International Conference on, 2014. IEEE, 1-6.

SONCHAN, P. & RAMINGWONG, S. ARMI 2.0: An online risk management simulation. Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2015 12th International Conference on, 2015. IEEE, 1-5.

ŠPUNDAK, M. 2014. Mixed agile/traditional project management methodology–reality or illusion? *Procedia-Social and Behavioral Sciences,* 119**,** 939-948.

STACEY, R. D. 2007. *Strategic management and organisational dynamics: The challenge of complexity to ways of thinking about organisations*, Pearson education.

STACH, W., KURGAN, L., PEDRYCZ, W. & REFORMAT, M. 2005. Genetic learning of fuzzy cognitive maps. *Fuzzy sets and systems,* 153**,** 371-401.

STANKOVIC, D., NIKOLIC, V., DJORDJEVIC, M. & CAO, D.-B. 2013. A survey study of critical success factors in agile software projects in former Yugoslavia IT companies. *Journal of Systems and Software,* 86**,** 1663-1678.

STEWART, K. J. & GOSAIN, S. 2006a. The impact of ideology on effectiveness in open source software development teams. *Mis Quarterly***,** 291-314.

STEWART, K. J. & GOSAIN, S. 2006b. The moderating role of development stage in free/open source software project performance. *Software Process: Improvement and Practice,* 11**,** 177-191.

STRAUSS, A. & CORBIN, J. 1990. *Basics of qualitative research,* Newbury Park, CA: Sage.

TAHERDOOST, H. & KESHAVARZSALEH, A. 2015a. Managing successful IT project; marketing perspective. *Romania,* 28**,** 11.

TAHERDOOST, H. & KESHAVARZSALEH, A. A Theoretical Review on IT Project Success/Failure Factors and Evaluating the Associated Risks. International Conference on Telecommunications and Informatics. Sliema, Malta, 2015b.

TAO, Y., HAN, D. & KIM, S. Writing Acceptable Patches: An Empirical Study of Open Source Project Patches. ICSME, 2014. 271-280.

TASHAKKORI, A. & TEDDLIE, C. 1998. *Mixed methodology: Combining qualitative and quantitative approaches,* Sage.

THAKURTA, R. 2013. Impact of Scope Creep on Software Project Quality. *Vilakshan: The XIMB Journal of Management,* 10.

THAKURTA, R. & SQUARE, X. 2012. Investigating Dynamics of Software Project Quality under Requirement Volatility Patterns. *Asian Journal of Information Technology,* 11**,** 186-193.

THOMAS, G. & FERNÁNDEZ, W. 2008. Success in IT projects: A matter of definition? *International Journal of Project Management,* 26**,** 733-742.

THUMMALAPENTA, S., CERULO, L., AVERSANO, L. & DI PENTA, M. 2010. An empirical study on the maintenance of source code clones. *Empirical Software Engineering,* 15**,** 1-34.

TIWANA, A. 2004. An empirical study of the effect of knowledge integration on software development performance. *Information and Software Technology,* 46**,** 899-906.

TIWANA, A. & KEIL, M. 2004. The one-minute risk assessment tool. *Communications of the ACM,* 47**,** 73-77.

TOMER, A. Software mangineeringment: teaching project management from software engineering perspective. Global Engineering Education Conference (EDUCON), 2014 IEEE, 2014. IEEE, 1-7.

TOO, E. G. & WEAVER, P. 2014. The management of project management: A conceptual framework for project governance. *International Journal of Project Management,* 32**,** 1382-1394.

TRENDOWICZ, A. & JEFFERY, R. 2014a. Challenges of Predictable Software Development. *Software Project Effort Estimation.* Springer.

TRENDOWICZ, A. & JEFFERY, R. 2014b. Common Factors Influencing Software Project Effort. *Software Project Effort Estimation.* Springer.

UZZAFER, M. 2013. A Risk Classification Scheme for Software Projects. *International Journal of Software Engineering and its Applications,* 7.

VADLAMANI, A., MANJULA, R. & SHEKHAR, D. 2016. Critical Facets of Project Management Triggering Major Effects on Software Development. *Imperial Journal of Interdisciplinary Research,* 3.

VARAJÃO, J., DOMINGUEZ, C., RIBEIRO, P. & PAIVA, A. 2014. Critical success aspects in project management: similarities and differences between the construction and software industry. *Tehnički vjesnik,* 21**,** 583-589.

VASILEIADOU, E. & SAFARZYŃSKA, K. 2010. Transitions: Taking complexity seriously. *Futures,* 42**,** 1176-1186.

VEHOVAR, V. & MANFREDA, K. L. 2008. Overview: online surveys. *The SAGE handbook of online research methods***,** 177-194.

VERNER, J., SAMPSON, J. & CERPA, N. What factors lead to software project failure? Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on, 2008a. IEEE, 71-80.

VERNER, J., SAMPSON, J. & CERPA, N. What factors lead to software project failure? 2008 Second International Conference on Research Challenges in Information Science, 2008b. IEEE, 71-80.

VERNER, J. M. & ABDULLAH, L. M. 2012. Exploratory case study research: Outsourced project failure. *Information and Software Technology,* 54**,** 866-886.

VERNER, J. M., BRERETON, O. P., KITCHENHAM, B. A., TURNER, M. & NIAZI, M. 2014. Risks and risk mitigation in global software development: A tertiary study. *Information and Software Technology,* 56**,** 54-78.

VIDAL, L.-A. & MARLE, F. 2008. Understanding project complexity: implications on project management. *Kybernetes,* 37**,** 1094-1110.

VIDAL, L.-A., MARLE, F. & BOCQUET, J.-C. Modelling project complexity. Proceedings of the International Conference on Engineering Design, ICED, 2007. 1-10.

VOSOUGH, A. 2012. Role of risk factor in project management.

VRHOVEC, S. L., HOVELJA, T., VAVPOTIČ, D. & KRISPER, M. 2015. Diagnosing organizational risks in software projects: Stakeholder resistance. *International Journal of Project Management,* 33**,** 1262-1273.

WALLACE, L., KEIL, M. & RAI, A. 2004. Understanding software project risk: a cluster analysis. *Information & Management,* 42**,** 115-125.

WALSTROM, K. A. & WILSON, R. L. 1997. An examination of executive information system (EIS) users. *Information & Management,* 32**,** 75-83.

WANG, J., LIN, W. & HUANG, Y.-H. 2010. A performance-oriented risk management framework for innovative R&D projects. *Technovation,* 30**,** 601-611.
WATERIDGE, J. 1995. IT projects: a basis for success. *International journal of project management,* 13**,** 169-172.
WATERIDGE, J. 1998. How can IS/IT projects be measured for success? *International journal of project management,* 16**,** 59-63.
WATT, S., SIMPSON, C., MCKILLOP, C. & NUNN, V. 2002. Electronic course surveys: does automating feedback and reporting give better results? *Assessment & Evaluation in Higher Education,* 27**,** 325-337.
WEBER, R. P. 1990. *Basic content analysis*, Sage.
WEERASINGHE, A., NANAYAKKARA, S., DISSANAYAKE, D., PERERA, S., ABEYSIRI, L. & SURAWEERA, T. 2014. Role of Change Management in Information System Implementations: Case of Custom Designed Information System Implementations in Sri Lanka.
WERNHAM, R. 1985. OBSTACLES TO STRATEGY IMPLEMENTATION IN A NATIONALIZED INDUSTRY [1]. *Journal of Management Studies,* 22**,** 632-648.
WESTERVELD, E. 2003. The Project Excellence Model®: linking success criteria and critical success factors. *International Journal of project management,* 21**,** 411-418.
WESTFALL, L. Software risk management. ASQ World Conference on Quality and Improvement Proceedings, 2000. American Society for Quality, 32.
WIKLUND, D. & PUCCIARELLI, J. 2009. Improving IT project outcomes by systematically managing and hedging risk. *An IDC Insights Research Document*.
WILDENBERG, M., BACHHOFER, M., ADAMESCU, M., DE BLUST, G., DIAZ-DELGADOD, R., ISAK, K., SKOV, F. & VARJOPURO, R. Linking thoughts to flows-Fuzzy cognitive mapping as tool for integrated landscape modelling. Proceedings of the 2010 International Conference on integrative landscape modeling: linking environmental, social and computer science, 2010. Citeseer, 5.
WILLIAMS, R., AMBROSE, K. & BENTREM, L. 2004. A Roadmap of Risk Diagnostic Methods: Developing an Integrated View of Risk Identification and Analysis Techniques. DTIC Document.
WILLIAMS, R. C. 2008. Identifying Program Risks. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.
WILLIAMS, R. C., PANDELIOS, G. J. & BEHRENS, S. G. 1999. *Software Risk Evaluation (SRE) Method Description: Version 2.0*, Carnegie Mellon University, Software Engineering Institute.
WILSON, M. & HOWCROFT, D. 2002. Re-conceptualising failure: social shaping meets IS research. *European Journal of Information Systems,* 11**,** 236-250.
WRIGHT, B. & SCHWAGER, P. H. 2008. Online survey research: can response factors be improved? *Journal of Internet Commerce,* 7**,** 253-269.
XIA, B. & CHAN, A. P. 2012. Measuring complexity for building projects: a Delphi study. *Engineering, Construction and Architectural Management,* 19**,** 7-24.
XIA, W. & LEE, G. 2005. Complexity of information systems development projects: conceptualization and measurement development. *Journal of management information systems,* 22**,** 45-83.
YANG, R., BAI, Y., QIN, Z. & YU, T. 2014. EgoNet: identification of human disease ego-network modules. *BMC genomics,* 15**,** 314.
YEW WONG, K. 2005. Critical success factors for implementing knowledge management in small and medium enterprises. *Industrial Management & Data Systems,* 105**,** 261-279.
YU, C.-P., CHEN, H.-G., KLEIN, G. & JIANG, R. 2015. The roots of executive information system development risks. *Information and Software Technology,* 68**,** 34-44.

ZHANG, H., KITCHENHAM, B. & JEFFERY, R. Planning software project success with semi-quantitative reasoning. Software Engineering Conference, 2007. ASWEC 2007. 18th Australian, 2007. IEEE, 369-378.

ZOWGHI, D. & COULIN, C. 2005. Requirements elicitation: A survey of techniques, approaches, and tools. *Engineering and managing software requirements.* Springer.

Appendix A

Questionnaire

.

A. Construct correlation questionnaire:

B. Rate the importance of **risk factors** on the failure of software project? And what is the impact of the following success criteria on the; **Cost control**, **project delivery Time**, **Quality and Scope**?

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
1.	Unclear customer requirements																					
2.	Unable to meet user requirements																					
3.	Lack of technical skills																					
4.	Technical complexity																					
5.	Low software performance																					
6.	Requirement creep																					
7.	Inappropriate development process/ methodology																					
8.	Problems with new technology																					
9.	Inadequate infrastructure																					

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
10	Unrealistic schedule																					
11	Unrealistic resource planning																					
12	Communication gaps																					
13	Conflicts among team members																					
14	Inefficient team capability																					
15	Staff turnover																					
16	Unrealistic budget																					
17	Resource insufficiency																					
18	User resistance																					
19.	Lack of law enforcement																					
20.	Understanding problems of customers																					
21	Understanding problem of developers																					
22	Improper planning																					
23	Project manager lacks experience																					
24	Government factors																					

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
25	Cultural diversity																					
26	Lack of motivation																					
27	Extensive personnel hiring																					
28	Inappropriate design																					
29	Inappropriate technology																					
30	Market demand obsolete																					
31	Improper marketing techniques																					
32	Lack of top management commitment and support																					
33	Size of the project																					
34	Incompatible development environment																					
35.	Unavailable customer contact																					
36.	Problems in testing tools																					
37	Gold plating																					
38	Developing the wrong software functions																					
39	Subcontracting																					
40	lack of project delivery milestones																					

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
41	Lack of staff experience																					
42	Backup issues																					
43	Natural disasters																					
44	Less reusability																					
45	Excessive error detection																					
46	Architecture complexity																					
47	Design is skipped or is created after code is written																					
48	Inconsistent coding style																					
49.	Lack of adequate security technologies (e.g., firewalls, encryption, etc.)																					
50.	Inadequate management of change																					
51	Data privacy issues																					
52	Insufficient consideration of reliability/availability																					
53	Insufficient consideration of system reset approach																					
54	No update plan to the final software product																					
55	No IP protection plan ,No version control																					

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
56	No backward compatibility and version management Plan																					
57	Use of cheap tools (software components, etc.)																					
58	Insufficient consideration of security and safety																					
59	Risk of problems with external tools and components																					
60	No training for managing outsource relationships																					
61	Lack of mechanism for validation and verification																					
62.	Unclear or misunderstood scope/objectives																					
63	Inadequate knowledge/non-technical skills																					
64	Project distribution																					
56	No backward compatibility and version management Plan																					
57	Use of cheap tools (software components, etc.)																					
58	Insufficient consideration of security and safety																					
59	Risk of problems with external tools and components																					

Number	Risk Factors	Rate the importance of risk factors on the software project.					Project success criteria															
							The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		Not important	Slightly important	Moderately important	Very important	Extremely important	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
60	No training for managing outsource relationships																					
61	Lack of mechanism for validation and verification																					
62.	Unclear or misunderstood scope/objectives																					
63	Inadequate knowledge/non-technical skills																					
64	Project distribution																					

What are the impact of the following project **success factors** on the **Cost control**, **project delivery Time**, **Quality and Scope**?

Number	Success Factors	Project success criteria															
		The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
1.	Clear requirements and specifications																
2.	Clear objectives and goals																
3.	Realistic schedule																
4.	Efficient project management																
5.	Top level management support																
6.	User/client involvement																
7.	Effective communication and feedback																
8.	Realistic budget																
9.	Skilled and sufficient Staff																
10.	Familiarity with technology																

Number	Success Factors	Project success criteria															
		The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
11	Appropriate development processes/ methodologies																
12	Proper planning																
13	Up-to-date progress reporting																
14	Effective monitoring and control																
15	Adequate resources																
16	Good leadership																
17	Risk management																
18	Change management																
19.	Appropriate infrastructure																
20.	Committed and motivated team																
21	Good quality management																
22	Managing the complexity of project size, number of organisations involved																
23	Pilot project performance																
24	Clear assignment of roles and responsibilities																

Number	Success Factors	Project success criteria															
		The Impact on project cost control				The Impact on project delivery time				The Impact on the Quality				The Impact on the scope			
		N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High	N/A	Low	Moderate	High
25	Team environment																
26	Customer training and education																
27	Efficient contract management																
28	Good performance by vendors/ contractors/ consultants																
29	Team capability																
30	Political stability																
31	Organizational culture																
32	Stability of organizational environment																
33	Project criticality																
34	Getting code from quality reliable and stable community																
35	Extensive testing for quality and careful selection of code																
36	Commitment of stakeholders																
37	Team training																
38	Project manager dedicated to the project																

B. Dependency matrix questionnaire

	Unclear customer requirements	Unable to meet user requirements	Lack of technical skills	Technical complexity	Low software performance	Requirement creep	Inappropriate development process/ methodology	Problems with new technology	Lack of motivation	Extensive personnel hiring	Inappropriate design	Inappropriate technology	Cost	Quality	Time	Scope
Clear requirements and specifications	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear objectives and goals	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Realistic schedule	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Efficient project management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Top level management support	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
User/client involvement	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Effective communication and feedback	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Realistic budget	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Skilled and sufficient Staff	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Familiarity with technology	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Appropriate development processes/ methodologies	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Proper planning	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Up-to-date progress reporting	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Effective monitoring and control	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Adequate resources	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Good leadership	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Risk management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Change management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Appropriate infrastructure	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Committed and motivated team	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Good quality management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Complexity, project size, number of organisations involved	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Pilot project performance	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear assignment of roles and responsibilities	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Team environment	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Customer training and education	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Efficient contract management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Good performance by vendors/ contractors/ consultants	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Team capability	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Political stability	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Organizational culture	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Stability of organizational environment	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Project criticality	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Getting code from quality reliable and stable community	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Extensive testing for quality and careful selection of code	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Commitment of stakeholders	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Team training	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Project manager dedicated to the project	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

	Size of the project	Incompatible development environment	Unavailable customer contact	Problems in testing tools	Gold plating	Developing the wrong software functions	Subcontracting	lack of project delivery milestones	Lack of staff experien ce	Backup issues	Natural disasters	Less reusability	Excessive error detection	Architecture complexity	Design is skipped or is created after code is written	Market demand obsolete	Improper marketing techniques	Lack of top management commitment and support
Clear requirements and specifications	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear objectives and goals	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Realistic schedule	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Efficient project management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Top level management support	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
User/client involvement	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Effective communication and feedback	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Realistic budget	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Skilled and sufficient Staff	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Familiarity with technology	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Appropriate development processes/ methodologies	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Proper planning	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Up-to-date progress reporting	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Effective monitoring and control	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Adequate resources	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Good leadership	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Risk management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Change management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Appropriate infrastructure	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Committed and motivated team	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Good quality management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Complexity, project size, number of organisations involved	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Pilot project performance	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear assignment of roles and responsibilities	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Team environment	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Customer training and education	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Efficient contract management	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Good performance by vendors/ contractors/ consultants	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Team capability	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Political stability	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Organizational culture	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Stability of organizational environment	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Project criticality	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Getting code from quality reliable and stable community	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Extensive testing for quality and careful selection of code	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Commitment of stakeholders	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Team training	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Project manager dedicated to the project	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

	Inconsistent coding style	Lack of adequate security technologies	Inadequate management of change	Data privacy issues	Insufficient consideration of reliability/ availability	Insufficient consideration of system reset approach	No update plan to the final software product	No IP protection plan ,No version control	No backward compatibility and version management Plan	Use of cheap tools (software components, etc.)	Insufficient consideration of security and safety	Risk of problems with external tools and components	No training for managing outsource relationships	Lack of mechanism for validation and verification	Unclear or misunderstood scope/ objectives	Project distribution
Clear requirements and specifications	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear objectives and goals	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Realistic schedule	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Efficient project management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Top level management support	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
User/client involvement	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Effective communication and feedback	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Realistic budget	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Skilled and sufficient Staff	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Familiarity with technology	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Appropriate development processes/ methodologies	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Proper planning	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Up-to-date progress reporting	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Effective monitoring and control	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Adequate resources	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Good leadership	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Risk management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Change management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Appropriate infrastructure	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Committed and motivated team	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Good quality management	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Complexity, project size, number of organisations involved	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Pilot project performance	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Clear assignment of roles and responsibilities	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Team environment	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Customer training and education	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Efficient contract management	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Good performance by vendors/ contractors/ consultants	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Team capability	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Political stability	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Organizational culture	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Stability of organizational environment	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Project criticality	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Getting code from quality reliable and stable community	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Extensive testing for quality and careful selection of code	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Commitment of stakeholders	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Team training	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Project manager dedicated to the project	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Cost	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Quality	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Time	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Scope	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Appendix B

<u>Tables</u>

Table 0-1 Statistical ranking results for all risk factors

Ref	Mean	Standard deviation	Management team	Development team	Coefficient of variation	Severity index	Overall Ranking
R1ENG	4.56	0.675	1	1	14.809226	91.214953	1
R20ENG	4.36	0.78	7	2	17.911853	87.102804	2
R22DEV	4.32	0.831	3	6	19.245683	86.35514	3
R41PR	4.31	0.905	4	8	21.006122	86.168224	4
R32DEV	4.31	0.915	6	4	21.246688	86.168224	5
R16PR	4.3	0.871	5	11	20.261976	85.981308	6
R14DEV	4.3	0.838	8	9	19.491567	85.981308	7
R23DEV	4.29	0.911	2	16	21.242499	85.794393	8
R2ENG	4.24	0.92	13	5	21.67845	84.859813	9
R18PR	4.21	1	19	3	23.73133	84.299065	10
R3ENG	4.21	0.932	9	21	22.109582	84.299065	11
R35DEV	4.21	0.89	11	18	21.12705	84.299065	12
R47DEV	4.21	0.939	10	19	22.329031	84.11215	13
R9DEV	4.17	0.947	12	23	22.708774	83.364486	14

R62ENG	4.17	0.916	22	10	21.979741	83.364486	15
R10DEV	4.17	0.874	20	12	20.968305	83.364486	16
R29ENG	4.16	0.779	21	13	18.727214	83.17757	17
R12DEV	4.15	0.93	24	17	22.403369	82.990654	18
R17PR	4.1	0.91	14	29	22.191393	82.056075	19
R30PR	4.1	1.063	17	28	25.919108	82.056075	20
R45ENG	4.09	0.986	16	31	24.088692	81.869159	21
R49ENG	4.08	1.047	28	22	25.640872	81.682243	22
R11DEV	4.08	0.933	25	25	22.841052	81.682243	23
R51PR	4.07	0.997	18	33	24.471946	81.495327	24
R6ENG	4.07	0.78	44	7	19.193889	81.308411	25
R52ENG	4.07	1.049	23	32	25.791876	81.308411	26
R28ENG	4.06	0.909	15	41	22.420776	81.121495	27
R27DEV	4.06	0.92	29	27	22.675093	81.121495	28
R26DEV	4.06	0.94	38	14	23.175357	81.121495	29
R13DEV	4.05	1.085	27	30	26.809684	80.934579	30
R15PR	4.04	0.921	43	15	22.803802	80.747664	31
R34ENG	4.02	0.961	35	26	23.92173	80.373832	32

R31ENG	3.99	1.112	30	39	27.85628	79.813084	33
R21ENG	3.99	0.818	41	24	20.507067	79.813084	34
R5ENG	3.99	0.986	31	35	24.70028	79.813084	35
R36ENG	3.97	1.032	26	49	25.984804	79.439252	36
R33ENG	3.96	1.063	49	20	26.833965	79.252336	37
R7DEV	3.95	0.935	32	47	23.663867	79.065421	38
R57ENG	3.93	1.021	33	46	25.954491	78.691589	39
R40DEV	3.92	0.982	42	40	25.08035	78.317757	40
R56DEV	3.91	1.086	37	43	27.805018	78.130841	41
R38DEV	3.91	0.976	34	52	24.995156	78.130841	42
R46ENG	3.9	1.027	47	38	26.360277	77.943925	43
R61ENG	3.89	1.067	39	48	27.441557	77.757009	44
R42DEV	3.89	1.119	36	56	28.773853	77.757009	45
R50DEV	3.88	0.978	45	44	25.220039	77.570093	46
R44ENG	3.85	1.035	54	34	26.886674	77.009346	47
R19PR	3.85	1.097	51	36	28.495199	77.009346	48
R60DEV	3.84	1.065	52	37	27.734488	76.82243	49
R54DEV	3.83	1.041	46	53	27.179594	76.635514	50

R53ENG	3.83	1.068	48	55	27.879781	76.635514	51
R63PR	3.81	1.074	53	45	28.165396	76.261682	52
R55DEV	3.79	1.026	50	54	27.026931	75.88785	53
R58ENG	3.75	1.182	40	61	31.549422	74.953271	54
R4ENG	3.74	0.904	56	50	24.188699	74.766355	55
R59PR	3.7	1.048	57	57	28.318349	74.018692	56
R39PR	3.69	1.085	60	51	29.395447	73.831776	57
R48ENG	3.67	1.106	62	42	30.099505	73.457944	58
R8DEV	3.67	1.035	58	58	28.179582	73.457944	59
R43PR	3.64	1.283	59	59	35.202197	72.897196	60
R64DEV	3.6	1.288	55	63	35.783289	71.962617	61
R24PR	3.56	1.199	61	60	33.669326	71.214953	62
R37DEV	3.45	1.159	63	62	33.61724	68.971963	63
R25DEV	2.99	1.349	64	64	45.117913	59.813084	64

Table 0-2 T-test analysis for rating the importance of risk factors in software project

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	t	95% Confidence Interval of the Difference	
									Lower	Upper
R1ENG	Equal variances assumed	3.833	.053	.838	105	.404	.110	.131	-.151	.371
	Equal variances not assumed			.815	86.247	.417	.110	.135	-.158	.379
R2ENG	Equal variances assumed	.223	.638	.773	105	.441	.138	.179	-.217	.494
	Equal variances not assumed			.771	99.916	.442	.138	.179	-.218	.494
R3ENG	Equal variances assumed	1.093	.298	1.970	105	.052	.352	.179	-.002	.706
	Equal variances not assumed			1.911	84.596	.059	.352	.184	-.014	.718
R6ENG	Equal variances assumed	.353	.554	-1.213	105	.228	-.184	.151	-.484	.116
	Equal variances not assumed			-1.221	102.679	.225	-.184	.150	-.482	.115

R28ENG	Equal variances assumed	6.035	.016	2.564	105	.012	.442	.172	.100	.783
	Equal variances not assumed			2.463	78.737	.016	.442	.179	.085	.799
R29ENG	Equal variances assumed	.090	.764	.654	105	.515	.099	.152	-.202	.400
	Equal variances not assumed			.642	91.465	.523	.099	.155	-.208	.406
R33ENG	Equal variances assumed	.183	.669	-.509	105	.612	-.106	.207	-.517	.306
	Equal variances not assumed			-.511	102.196	.610	-.106	.206	-.515	.304
R45ENG	Equal variances assumed	4.312	.040	1.893	105	.061	.358	.189	-.017	.734
	Equal variances not assumed			1.840	85.791	.069	.358	.195	-.029	.746
R62ENG	Equal variances assumed	1.070	.303	.439	105	.662	.078	.179	-.276	.433
	Equal variances not assumed			.427	86.942	.670	.078	.184	-.286	.443
R20ENG	Equal variances assumed	1.284	.260	1.008	105	.316	.153	.152	-.148	.454
	Equal variances			.986	89.504	.327	.153	.155	-.155	.461

	not assumed									
R9DEV	Equal variances assumed	.938	.335	1.672	105	.097	.305	.182	-.057	.667
	Equal variances not assumed			1.619	83.573	.109	.305	.188	-.070	.680
R10DEV	Equal variances assumed	3.960	.049	.682	105	.497	.116	.170	-.222	.454
	Equal variances not assumed			.664	87.003	.508	.116	.175	-.231	.464
R12DEV	Equal variances assumed	1.340	.250	.663	105	.509	.120	.181	-.239	.479
	Equal variances not assumed			.650	90.773	.518	.120	.185	-.247	.487
R14DEV	Equal variances assumed	1.846	.177	1.722	105	.088	.278	.161	-.042	.598
	Equal variances not assumed			1.667	83.309	.099	.278	.167	-.054	.609
R22DEV	Equal variances assumed	1.738	.190	1.712	105	.090	.274	.160	-.043	.591
	Equal variances not assumed			1.651	80.969	.103	.274	.166	-.056	.604

R23DEV	Equal variances assumed	.950	.332	2.149	105	.034	.374	.174	.029	.720
	Equal variances not assumed			2.088	85.512	.040	.374	.179	.018	.731
R26DEV	Equal variances assumed	2.624	.108	-.476	105	.635	-.087	.183	-.451	.276
	Equal variances not assumed			-.467	91.272	.642	-.087	.187	-.458	.284
R32DEV	Equal variances assumed	.169	.682	1.452	105	.149	.257	.177	-.094	.608
	Equal variances not assumed			1.433	94.309	.155	.257	.179	-.099	.613
R35DEV	Equal variances assumed	.470	.494	1.609	105	.111	.276	.172	-.064	.617
	Equal variances not assumed			1.577	90.560	.118	.276	.175	-.072	.625
R47DEV	Equal variances assumed	1.517	.221	1.642	105	.104	.297	.181	-.062	.656
	Equal variances not assumed			1.608	90.394	.111	.297	.185	-.070	.665
R15PR	Equal variances assumed	.003	.954	-.675	105	.501	-.121	.179	-.477	.235
	Equal variances			-.678	101.941	.500	-.121	.179	-.476	.233

	not assumed									
R16PR	Equal variances assumed	3.522	.063	1.887	105	.062	.316	.167	-.016	.647
	Equal variances not assumed			1.791	71.493	.078	.316	.176	-.036	.667
R17PR	Equal variances assumed	.006	.936	1.932	105	.056	.338	.175	-.009	.684
	Equal variances not assumed			1.905	94.130	.060	.338	.177	-.014	.689
R18PR	Equal variances assumed	.263	.609	.061	105	.951	.012	.195	-.375	.399
	Equal variances not assumed			.062	101.739	.951	.012	.195	-.374	.398
R30PR	Equal variances assumed	.183	.670	1.458	105	.148	.300	.206	-.108	.707
	Equal variances not assumed			1.452	98.987	.150	.300	.206	-.110	.709
R41PR	Equal variances assumed	2.587	.111	1.691	105	.094	.295	.174	-.051	.641
	Equal variances not assumed			1.634	82.426	.106	.295	.180	-.064	.654

R51PR	Equal variances assumed	5.200	.025	1.689	105	.094	.325	.192	-.057	.706
	Equal variances not assumed			1.634	83.153	.106	.325	.199	-.070	.719